Journal of Logical and Algebraic Methods in Programming 85 (2016) 1011–1037

Contents lists available at ScienceDirect

Journal of Logical and Algebraic Methods in Programming

www.elsevier.com/locate/jlamp

An exercise on the generation of many-valued dynamic logics

Alexandre Madeira [a,*], Renato Neves [a], Manuel A. Martins [b]

[a] INESC TEC (HASLab) & Minho University, Portugal
[b] CIDMA – Dep. of Mathematics, Aveiro University, Portugal

ARTICLE INFO

Article history:
Received 7 July 2015
Received in revised form 24 March 2016
Accepted 30 March 2016
Available online 4 April 2016

Keywords:
Dynamic logic
Many-valued logic
Kleene algebra
Action lattice

ABSTRACT

In the last decades, dynamic logics have been used in different domains as a suitable formalism to reason about and specify a wide range of systems. On the other hand, logics with many-valued semantics are emerging as an interesting tool to handle devices and scenarios where uncertainty is a prime concern. This paper contributes towards the combination of these two aspects through the development of a method for the systematic construction of many-valued dynamic logics. Technically, the method is parameterised by an action lattice that defines both the computational paradigm and the truth space (corresponding to the underlying Kleene algebra and residuated lattices, respectively).

© 2016 Elsevier Inc. All rights reserved.

1. Introduction

1.1. Context

Propositions, capturing static properties of program states, and events, or actions, standing for state transitions, are the key ingredients in modelling and analysing about state-based software systems. Programs are typically combined through a Kleene algebra to express sequential, non deterministic, iterative behaviours, while propositions bring to the scene a logical structure.

Dynamic logic [17], a generalisation of the logic of Floyd–Hoare, is a well known and particularly powerful way of combining these two dimensions into a formal framework to reason about computational systems. Its potential stems from blending together classical logic, enriched with a modal dimension to express system's dynamics, and a (Kleene) algebra of actions to structure programs.

Over time dynamic logic grew to an entire family of logics increasingly popular in the verification of computational systems, and able to evolve and adapt to new, and complex validation challenges. One could mention its role in model validation (as in *e.g.* [26]), or the whole family of variants tailored to specific programming languages (as in *e.g.* [32,2]), or its important extensions to new computing domains, namely probabilistic [20] or continuous [38,39].

The latter is particularly relevant from an Engineering point of view: Actually, Platzer's hybrid dynamic logic, and its associated tool, KeYmaera, combining an algebra of actions based on real numbers assignments, with the standard Kleene operators and differential equations to specify continuous transitions from the "real" (physical) world, provides a powerful framework with increased industrial relevance for the design and validation of hybrid systems.

If hybrid systems entail the need to handle continuous state spaces, in a number of other cases dealing with some form of "quantitative" transitions (weighted, costed, probabilistic, certainty degrees etc.) is also a must. This motivates

* Corresponding author.
E-mail addresses: amadeira@inestec.pt (A. Madeira), rjneves@inestec.pt (R. Neves), martins@ua.pt (M.A. Martins).

http://dx.doi.org/10.1016/j.jlamp.2016.03.004
2352-2208/© 2016 Elsevier Inc. All rights reserved.

research to define dynamic logics over structures able to model weighted computations. On the logical side, expressing the validity of a formula through a Boolean outcome can be also be quite restrictive when dealing with complex, often unpredictable, systems. This motivates the adoption of logics with many-valued semantics, *e.g.* fuzzy [15], probabilistic [33] or weighted [10].

In such a context, this work attempts to combine dynamic logic and many-valued semantics to capture smoothly these kind of phenomena. The first steps in this direction appeared in a previous conference paper [29] where a generic, parametric, method to construct (propositional) many-valued dynamic logics was discussed. Technically, the definition of these logics is parameterised by an action lattice [21] which combines a Kleene algebra with a residuated lattice structure. This algebraic structure fits well our goal. On the one hand, as a residuated lattice, it provides an abstract structure for the truth spaces – most of the semantic structures used as truth spaces, such as *Boolean algebras*, *Heyting algebras*, *MV algebras* or *Łukasiewicz arithmetic lattices*, are residuated lattices (*e.g.*, [14,19]). Here, the residues take the role of logic implication. On the other hand as a Kleene algebra, it provides an abstract, generic, model for computations (*e.g.*, probabilistic, weighted,...). Moreover the extension of Kleene algebras with a residuated operator, providing a left inverse to sequential composition as in [40], as well as with a lattice structure, leads to a finitely-based equational variety which, as plain Kleene algebras, is closed under the formation of square matrices [22]. The relevance of this closure property lies in the fact that several problems modelled as (weighted) transition systems can be formulated as matrices over a Kleene algebra or a related structure. Following such a trend, we represent programs as matrices supporting the information about their effects when executed from each state in the state space.

1.2. Contributions

This paper is part of a research agenda on a systematic development of dynamic logics.[1] In particular, it extends preliminary results documented in [29] in several directions. On the one hand, the scope of the parameters is generalised by taking arbitrary action lattices instead of just the integral ones. The expressiveness of the dynamisation process is also strongly enriched, through the consideration of negations, [_]-modalities and tests. On the other hand, we took the challenge here to characterise, in some sense, "*how dynamic dynamisations are?*". The issue is addressed through an exercise: assuming the axiomatics of propositional dynamic logic (PDL) as a reference, we carry on a systematic study of the validity of some of its particular fragments with respect to particular dynamisation classes. Note, that this criteria is not an absolute reference to judge what is, and what is not, a dynamic logic. Although PDL is probably the most popular dynamic logic, others exist which do not satisfy such axiomatics. Such is the case, for example, of *game logic* [37].

To the best of our knowledge, beyond our preliminary work [29], the approaches reported in [18,25] are the unique references in the literature addressing many-valued dynamic logic. In the first paper J. Hughes *et al.* introduced a propositional dynamic logic over the continuum truth $(0, 1)$-lattice with the standard fuzzy residues (actually the ones adopted in Example 4). In particular, this logic can be achieved by weakening a specific instance built with the general construction introduced in the present paper. However, from the perspective of dynamic logic, this formalism is quite restrictive, since it lefts behind both transitive closure and non-deterministic choice.

In the context of rational decision theory, C. Liau [25] introduced a many-valued dynamic logic w.r.t. the specific continuum truth $(0, 1)$-lattice. Admitting some level of parametricity on the implication adopted (through a notion of implication function, of which the implications of Łukasiewicz and Gödel are examples), the semantics of [_]-modalities becomes quite different from what we get. By this reason, differently from what happens with the one introduced in [18], this logic can not be captured with our generic formalism.

More extensive work exist in the related field of many-valued modal logics. Two approaches are usually considered. The first one is clearly conservative, in the sense that the many-valued semantics only affects the modal valuation of propositions. In this case the accessibility relations are crispy (classic). The second one, closer to our own, considers that accessibility relations can themselves be many-valued. This approach was introduced by M. Fitting in [11,12], with many-valueness evaluated in finite Heyting algebras. Later it was deeply investigated by F. Bou *et al.* in [6], who adopted the more generic truth support of finite integral commutative residuated lattices. A middle-term between crispy and many-valued accessibility relations, appears in some works (*e.g.* [5,31]) through multi-modalities: for the cases where the truth lattice is a chain, any multi-valued relation can be equivalently expressed using a decreasing family of crispy modal relations, indexed by the support of the respective lattice.

1.3. A tribute to José Nuno Oliveira

The interplay between logic and computation, lying at the very heart of dynamic logic, is pervasive in the scientific work of José Nuno Oliveira. From his perspective, any computational phenomenon is an arrow in a suitable universe whose source and target are logic expressions. Thus, we find computations typed by invariants, as in a calculus of conductive programs [1], functions, in a calculus of data dependencies used for type checking database operations and query optimisation [36], or program assertions, in the form of coreflexive relations, as in a Hoare logic like calculus [34] targeting correct-by-construction program design, rather than verification.

[1] http://wiki.di.uminho.pt/twiki/bin/view/Research/Dali/WebHome.

$$a + (b + c) = (a + b) + c \quad (1)$$
$$a + b = b + a \quad (2)$$
$$a + a = a \quad (3)$$
$$a + 0 = 0 + a = a \quad (4)$$
$$a; (b; c) = (a; b); c \quad (5)$$
$$a; 1 = 1; a = a \quad (6)$$
$$a; (b + c) = (a; b) + (a; c) \quad (7)$$
$$(a + b); c = (a; c) + (b; c) \quad (8)$$
$$a; 0 = 0; a = 0 \quad (9)$$
$$1 + a + (a^*; a^*) \leq a^* \quad (10)$$
$$a; x \leq x \Rightarrow a^*; x \leq x \quad (11)$$
$$x; a \leq x \Rightarrow x; a^* \leq x \quad (12)$$
$$a; x \leq b \Leftrightarrow x \leq a \rightarrow b \quad (13)$$
$$a \rightarrow b \leq a \rightarrow (b + c) \quad (14)$$
$$(x \rightarrow x)^* = x \rightarrow x \quad (15)$$
$$x \leq a \rightarrow (a; x) \quad (16)$$
$$a \cdot (b \cdot c) = (a \cdot b) \cdot c \quad (17)$$
$$a \cdot b = b \cdot a \quad (18)$$
$$a \cdot a = a \quad (19)$$
$$a + (a \cdot b) = a \quad (20)$$
$$a \cdot (a + b) = a \quad (21)$$
$$a; (a \rightarrow b) \leq b \quad (22)$$

Fig. 1. Axiomatisation of action lattices (from [21]).

In such calculi, universal coalgebra, the theory of functional dependencies or (generalised) Hoare logics, all rendered in a Tarskian, point-free style amenable to algebraic manipulation, play the role of type systems whose rules help in reasoning about computations without diving into their semantic intricacies. Moreover, they provide the basis of extended type checking mechanisms to discharge the relevant proof obligations.

José's work explores the power of abstract algebraic structures as a way to promote calculation in software design. Such structures, of which Kleene algebras are a prime example, endow techniques which have been shown to be amenable to automation. Moreover, they have the power to unify seemingly disparate domains and theories once the latter are encoded into the same abstract terms.

We believe the piece of research reported in this paper, seeking to formulate dynamic logics for weighted, multi-valued computations, goes in a similar direction. In particular, it may contribute to set the context for the challenge José identified in a landmark tutorial [34]:

> *The idea that the proposed calculus bridges Hoare logic and type theory needs to be better exploited, in particular concerning the work by Kozen on subsuming propositional Hoare logic under Kleene algebra with tests (of which the relational calculus is a well known instance).*

The generic nature of the approach taken here, in which logics are parameterised by specific kinds of action lattices, as well as the possible relevance of our results to the study of weighted transition systems (another of José's research interests [35]), are also a tribute to his work. So is the form of this paper, set as an *exercise*, which, as we have learned from José, is the most effective source of all generality.

1.4. Outline of the paper

The paper is organised as follows. Section 2, recalls what an action lattice is. The whole catalogue of technical properties used in the paper are proven. Then, an hierarchy of three classes of action lattices is introduced and characterised in detail. The parametric construction of many-valued dynamic logics is presented in Section 3. Both the generality of the computational models and semantics are illustrated in this section. The question "how dynamic are dynamisations?" is then addressed, leading to a systematic study on the validity of PDL fragments with respect to particular classes of action lattices. Finally, Section 4 concludes and enumerates elements for future work.

2. Action lattices

2.1. Definition and properties

Let us start by recalling from [21] the following definition:

Definition 1. An *action lattice* is a tuple

$$\mathbf{A} = (A, +, ;, 0, 1, *, \rightarrow, \cdot)$$

where A is a set, 0 and 1 are constants and $+, ;, *, \rightarrow$ and $\cdot$ are binary operations in A satisfying the axioms enumerated in Fig. 1, where the relation $\leq$ is induced by $+$: $a \leq b$ iff $a + b = b$.

Note that, by (20) and (21), the natural order $\leq$ can be equivalently defined by $a \leq b$ iff $a \cdot b = a$. Observe that by restricting the definition of **A** to the structure $(A, +, ;, 0, 1, *)$ axiomatised by (1)–(12) we obtain the definition of a *Kleene algebra* [8,22]. In the context of this work, this will be called the underlying Kleene algebra of **A**. Moreover, by considering structure $(A, +, ;, 0, 1, \rightarrow, *)$ axiomatised by (1)–(16) we obtain the definition of (left-residuated) *action algebra* [40]. The interested reader is referred to [13] for a detailed discussion on the relationship between Kleene algebras, action algebras and action lattices. As stated in the introduction, the structure of an action lattice is explored in this paper along a double dimension: as a computational model and as a truth space. The intuitions for some of its operations shall be taken from both of these perspectives. Such is the case of operation $+$, which plays the role of non-deterministic choice, in the interpretation of programs, and of logical disjunction, in the interpretation of sentences. However, there are operations whose intuition is borrowed from just in one of these domains. For instance, while operations $*$ and $;$ are taken as iterative application and sequential composition of actions, operations $\rightarrow$ and $\cdot$ play the role of logical implication and conjunction, respectively.

The following theorem establishes a set of well known Kleene algebra properties to be used in the sequel.

Theorem 1. *(See [8].) Let $(A, +, ;, 0, 1, *)$ be a Kleene algebra. The following properties hold:*

$$a \leq a^* \tag{23}$$

$$a^* = a^{**} \tag{24}$$

$$a^* = a^*; a^* \tag{25}$$

$$1 + a; a^* = a^* \tag{26}$$

An action lattice is said to be *complete* when any subset of A has both a supremum and an infimum w.r.t. $\leq$. The greatest and least elements, when they exist, are denoted in the sequel by $\top$ and $\bot$, respectively. Note that in any action lattice $\bot = 0$, since for any $a \in A$, $a + 0 = a$, i.e., $0 \leq a$. In the case of complete action lattices, the existence of greatest elements is also ensured. Since $+$ and $\cdot$ are associative, we can generalise them to n-ary operators, and we resort to notation $\sum$ for the iterated version of the (join) operator $+$, and $\prod$ for the iterated version of the (meet) operator $\cdot$ of action lattices. The left associativity of operators $+$ and $\cdot$ is, as usual, assumed. Differently from the original definition [21], where a pair of residues is considered, we keep only the left one. Equivalence $a \leftrightarrow b$ is understood as the value $(a \rightarrow b); (b \rightarrow a)$. Note that, as usual (*e.g.* [6]), the composition $;$, rather than meet $\cdot$, is taken as conjunction in the definition of $\leftrightarrow$. The reasons for this choice will become clear in the discussion of duality between box and diamond modalities (subsection 3.2.4). Actually, the adjoint of implication $\rightarrow$ is composition $;$ (and not the join $\cdot$). In the context of many-valued logics, the connective $;$ is called strong conjunction

As mentioned above, this structure supports both the computational paradigm (to distinguish between *e.g.* imperative, deterministic or non-deterministic computations, or between plain or weighted transitions) and the truth space (to capture *e.g.* the standard Boolean reasoning or more complex truth spaces).

The following Lemma provides a catalogue of useful properties of action lattices.

Lemma 1. *Let* **A** *be an action lattice.*
If **A** *is complete and I finite:*

$$x \rightarrow (\prod_{i \in I} y_i) = \prod_{i \in I} (x \rightarrow y_i) \tag{27}$$

$$(\sum_{i \in I} x_i) \rightarrow y = \prod_{i \in I} (x_i \rightarrow y) \tag{28}$$

$$a \cdot \top = a = \top \cdot a \tag{29}$$

$$(a \rightarrow \top) = \top \tag{30}$$

The following properties hold in any action lattice **A**:

$$x \leq y \Rightarrow a; x \leq a; y \tag{31}$$

$$x \leq y \Rightarrow x; a \leq y; a \tag{32}$$

$$a \leq b \,\&\, c \leq d \Rightarrow a + c \leq b + d \tag{33}$$

$$a \rightarrow (b \cdot c) \leq a \rightarrow b \tag{34}$$

$$a \leq b \,\&\, c \leq d \Rightarrow a \cdot c \leq b \cdot d \tag{35}$$

$$a; (b \cdot c) \leq (a; b) \cdot (a; c) \tag{36}$$

$$a \rightarrow (b \cdot c) \leq (a \rightarrow b) \cdot (a \rightarrow c) \tag{37}$$

$$a \leq b \Rightarrow (c \rightarrow a) \leq (c \rightarrow b) \tag{38}$$

$$a \leq b \Rightarrow (b \rightarrow c) \leq (a \rightarrow c) \tag{39}$$

$$a \rightarrow (b \rightarrow c) = (b; a) \rightarrow c \tag{40}$$

$$a \leq b \,\&\, a \leq c \Rightarrow a \leq b \cdot c \tag{41}$$

$$a \leq b \,\&\, c \leq d \Rightarrow a; c \leq b; d \tag{42}$$

$$1 \leq (0 \rightarrow a) \tag{43}$$

If $\mathbf{A}$ *is* $;$*-commutative,*

$$x; x = x \Rightarrow (x \to (y \to z)); (x \to y) \leq (x \to z) \tag{44}$$

When I finite, we have also,

$$\sum_{i \in I} (a_i \cdot b_i) \leq \sum_{i \in I} a_i \cdot \sum_{i \in I} b_i \tag{45}$$

Proof. For the proof of (27) and (28) see [6]. The remaining cases are proved in the sequel.

Equation (29) holds since $\top$ is the greatest element of the lattice, *i.e.*, that for any $a \in A$, $a \leq \top$.

To establish (30) we just have to note that, since $\top$ is the greatest element, we have $y \leq \top$ for any $y \in A$ and, in particular, a; $\top \leq \top$. Hence, by (13), $\top \leq a \to \top$. Since $\top$ is the greatest element, $(a \to \top) = \top$.

For (31), assuming $x \leq y$, *i.e.*, $x + y = y$. By (7). Then, $a; x + a; y = a; (x + y) = a; y$.

The proof for (32) is analogous but resorting to distributivity (8).

For (33), assuming $a \leq b$ and $c \leq d$, *i.e.*, $a+b = b$ and $c+d = d$. By (1) and (2), we have $(a+c)+(b+d) = (a+b)+(c+d) = b + d$.

In order to prove (34) we observe that

$$\begin{array}{ll|ll} & a \to x \leq a \to (x + b) & & \\ \Rightarrow & \quad \{ \text{ subst. } x \mapsto b \cdot c \ \} & \Leftrightarrow & \quad \{ \text{ (2) and (20)} \} \\ & a \to (b \cdot c) \leq a \to ((b \cdot c) + b) & & a \to (b \cdot c) \leq a \to b \end{array}$$

By (14) we have $a \to x \leq a \to (x + b)$. Hence $a \to (b \cdot c) \leq a \to b$ holds.

In order to prove (35), assuming $a \cdot b = a$ and $c \cdot d = c$ and applying (17) and (18) we have $(a \cdot c) \cdot (b \cdot d) = (a \cdot b) \cdot (c \cdot d) = a \cdot (c \cdot d) = a \cdot c$.

For (36), we observe that

$$\begin{array}{ll|ll} & b.c \leq b \ \text{ and } b.c \leq c & & a; (b.c) \cdot a; (b.c) \leq (a; b) \cdot (a; c) \\ \Leftrightarrow & \quad \{ (31) \} & \Leftrightarrow & \quad \{ (19) \} \\ & a; (b.c) \leq a; b \ \text{ and } a; (b.c) \leq a; c & & a; (b.c) \leq (a; b) \cdot (a; c) \\ \Leftrightarrow & \quad \{ (35) \} & & \end{array}$$

Since by properties for any $a, b, c \in \mathbf{A}$, $b.c \leq b$ and $b.c \leq c$, we have that $a; (b.c) \leq (a; b) \cdot (a; c)$.

In order to prove (37), we observe that

$$\begin{array}{ll|ll} & a \to (b \cdot c) \leq a \to b \text{ and} & & \\ & a \to (b \cdot c) \leq a \to c & & \\ \Rightarrow & \quad \{ (35) \} & \Leftrightarrow & \quad \{ (19) \} \\ & (a \to (b \cdot c)) \cdot (a \to (b \cdot c)) \leq & & (a \to (b \cdot c)) \leq \\ & (a \to b) \cdot (a \to c) & & (a \to b) \cdot (a \to c) \end{array}$$

By (34), we have a $\to (b \cdot c) \leq a \to b$ and $a \to (b \cdot c) \leq a \to c$. Therefore $(a \to (b \cdot c)) \leq (a \to b) \cdot (a \to c)$.

In order to prove (38), property (14) yields $(c \to a) \leq (c \to (a + b))$. By hypothesis $a \leq b$, *i.e.*, $a + b = b$. Thus, $(c \to a) \leq (c \to b)$.

In order to prove (39), we have by hypothesis that $a \leq b$, *i.e.*, $a+b = b$. Hence $(b \to c) = ((a+b) \to c)$. By (28), $((a+b) \to c) = (a \to c) \cdot (b \to c)$. Hence $(b \to c) = (a \to c) \cdot (b \to c)$, *i.e.*, $(b \to c) \leq (a \to c)$.

To establish (40) we reason

$$\begin{array}{ll|ll} & b \to c \leq b \to c & & \quad \{ (5) \} \\ \Leftrightarrow & \quad \{ (13) \} & & b; (a; ((b; a) \to c)) \leq c \\ & b; (b \to c) \leq c & \Leftrightarrow & \quad \{ (13) \} \\ \Rightarrow & \quad \{ \text{ subst. } b \mapsto b; a \} & & a; ((b; a) \to c) \leq b \to c \\ & (b; a); ((b; a) \to c) \leq c & \Leftrightarrow & \quad \{ (13) \} \\ & & & (b; a) \to c \leq a \to (b \to c) \end{array}$$

Moreover,

$$\begin{array}{ll|ll} & a \to b \leq a \to b & & \\ \Leftrightarrow & \{\ (13)\} & \Rightarrow & \{\ (22) \text{ and transitivity }\} \\ & a;(a \to b) \leq b & & b;(a;(a \to (b \to c))) \leq c \\ \Rightarrow & \{\ \text{subst. } b \mapsto b \to c\} & \Leftrightarrow & \{\ (5)\} \\ & a;(a \to (b \to c)) \leq b \to c & & (b;a);(a \to (b \to c)) \leq c \\ \Rightarrow & \{\ (31)\} & \Leftrightarrow & \{\ (13)\} \\ & b;(a;(a \to (b \to c))) \leq b;(b \to c) & & a \to (b \to c) \leq (b;a) \to c \end{array}$$

For the proof of (41) let us suppose $a \leq b$ and $a \leq c$, *i.e.*, that $a \cdot b = a$ and $a \cdot c = a$. Then, $(a \cdot b) \cdot c = a \cdot c = a$, *i.e.*, $a \leq b \cdot c$.

Regarding property (42) assume that $a \leq b$ and $c \leq d$. Then, by (7) and since $c \leq d \Leftrightarrow c + d = d$, we have $b;c + b;d = b;(c + d) = b;d$, *i.e.*, $b;c \leq b;d$. Moreover, by (8) and since $a \leq b \Leftrightarrow a + b = b$, we have $b;c = (a + b);c = a;c + b;c$, *i.e.*, $a;c \leq b;d$.

In order to prove (43), since 0 is the smallest element of the lattice, by (6), (32) and (13), we have that, $0 \leq a \Rightarrow 0;1 \leq a;1 \Rightarrow 0;1 \leq a \Leftrightarrow 1 \leq 0 \to a$.

For the proof of (44) let **A** be ;-commutative and let us assume that $x = x;x$.

$$\begin{array}{ll|ll} & ((x \to (y \to z));(x \to y));x & & \\ = & \{\ \text{since } x = x;x\} & \leq & \{\ (22) + (42)\} \\ & ((x \to (y \to z));(x \to y));(x;x) & & y;(y \to z) \\ = & \{\ (5) + \text{;-commutativity}\} & \leq & \{\ (22)\} \\ & (x;(x \to y));(x;(x \to (y \to z))) & & z \end{array}$$

Hence,

$$\begin{array}{ll|ll} & (x \to (y \to z));(x \to y);x \leq z & & \\ \Leftrightarrow & \{\ \text{;-commutativity}\} & \Leftrightarrow & \{\ (13)\} \\ & x;(x \to (y \to z));(x \to y) \leq z & & (x \to (y \to z));(x \to y) \leq (x \to z) \end{array}$$

In the proof of (45), supposing a finite I, and $a_i \cdot b_i \leq a_i$ and $a_i \cdot b_i \leq b_i$ we have by (33) that $\sum_{i\in I}(a_i \cdot b_i) \leq \sum_{i\in I} a_i$ and $\sum_{i\in I}(a_i \cdot b_i) \leq \sum_{i\in I} b_i$. Hence, by (41) $\sum_{i\in I}(a_i \cdot b_i) \leq \sum_{i\in I} a_i \cdot \sum_{i\in I} b_i$ holds. □

Please note that we extended some of the axioms for $+$ and $\cdot$ to the corresponding generalised versions, $\sum$ and $\prod$, respectively (*e.g.* $a_i \leq b_i \Rightarrow \sum_{i\in I} a_i \leq \sum_{i\in I} b_i$ stands for the generalised version of (33)). Generalised versions of (33), (35) and (41) are used in this way.

Now we discuss a number of illustrative examples. Among them, Examples 1, 2, 4, and 5 were already presented in [29].

Example 1 (**2** – *linear two-values lattice*). Let us consider the well known binary structure

$$\mathbf{2} = (\{\top, \bot\}, \vee, \wedge, \bot, \top, *, \to, \wedge)$$

with the standard boolean connectives:

$\vee$	$\bot$	$\top$
$\bot$	$\bot$	$\top$
$\top$	$\top$	$\top$

$\wedge$	$\bot$	$\top$
$\bot$	$\bot$	$\bot$
$\top$	$\bot$	$\top$

$\to$	$\bot$	$\top$
$\bot$	$\top$	$\top$
$\top$	$\bot$	$\top$

$*$	
$\bot$	$\top$
$\top$	$\top$

It is not difficult to see that **2** is an action lattice. Moreover, the action lattice is obviously complete and it satisfies condition (36) (note that both composition and meet are realised by $\wedge$).

Example 2 (**3** – *linear three-value lattice*). The explicit introduction of a denotation for *unknown* give rise to the following three elements linear lattice

$$\mathbf{3} = (\{\top, u, \bot\}, \vee, \wedge, \bot, \top, *, \to, \wedge)$$

where

$\vee$	$\bot$	u	$\top$
$\bot$	$\bot$	u	$\top$
u	u	u	$\top$
$\top$	$\top$	$\top$	$\top$

$\wedge$	$\bot$	u	$\top$
$\bot$	$\bot$	$\bot$	$\bot$
u	$\bot$	u	u
$\top$	$\bot$	u	$\top$

$\to$	$\bot$	u	$\top$
$\bot$	$\top$	$\top$	$\top$
u	$\bot$	$\top$	$\top$
$\top$	$\bot$	u	$\top$

$*$	
$\bot$	$\top$
u	$\top$
$\top$	$\top$

It is easy to see that all the conditions in Definition 1 hold. Moreover, the action lattice is complete.

Example 3 *(2^A – powerset of A).* For a fixed, finite set A, let us consider the structure

$$2^A = (\mathcal{P}(A), \cup, \cap, \emptyset, A, *, \rightarrow, \cap)$$

where $\mathcal{P}(A)$ denotes the powerset of A, $\cup$ and $\cap$ are set union and intersection operations, $*$ maps each set $X \in \mathcal{P}(A)$ into A and $X \rightarrow Y = X^c \cup Y$, where $X^c = \{x \in A \mid x \notin X\}$. This action lattice is also complete.

Example 4 *(**Ł** – the Łukasiewicz arithmetic lattice).* This example is based on the well-known Łukasiewicz arithmetic lattice

$$\mathbf{Ł} = ([0,1], max, \odot, 0, 1, *, \rightarrow, min)$$

where $x \rightarrow y = min\{1, 1 - x + y\}$, $x \odot y = max\{0, y + x - 1\}$ and $*$ maps each point of $[0,1]$ to 1. Again, this is a complete action lattice.

Example 5 *(**FW** – the Floyd–Warshall algebra).* The Floyd–Warshall algebra consists of a tuple

$$\mathbb{N}^+_{\perp\top} = (\{\perp, 0, 1, \ldots, \top\}, max, +, \perp, 0, *, \smile, min)$$

where $+$ extends addition on $\mathbb{N}$ by considering $\perp$ as its absorbent element and $a + \top = \top = \top + a$ for any $a \neq \perp$. Operation *max* (respectively, *min*) is defined as the maximum (respectively, minimum) under the order $\perp < 0 < \cdots < \top$. Operation $\smile$ is truncated subtraction

$$a \smile b = \begin{cases} \top, & \text{if } a = \perp \text{ or } b = \top \\ b - a, & \text{if } b \geq a \text{ and } a, b \in \mathbb{N} \\ 0, & \text{if } a > b \text{ and } a, b \in \mathbb{N} \\ \perp & \text{otherwise} \end{cases}$$

and, for any natural $i > 0$,

$*$	
$\perp$	0
0	0
i	$\top$
$\top$	$\top$

.

Note that the order induced by $a \leq b$ iff $max\{a, b\} = b$ corresponds to the one mentioned above. The action lattice is complete.

Example 6 *(**REL**(A)-relational algebra over a set A).* Let us consider the action lattice defined by relations over a set A. The corresponding Kleene algebra turns to be quite paradigmatic, since it underlies most standard semantics for sequential programs based on input/output relations.

Given a set A, we have

$$\mathbf{REL}(A) = (\mathcal{P}(A^2), \cup, \circ, \emptyset, \Delta, *, \backslash, \cap)$$

where $\cup$ and $\cap$ stand for set union and intersection, respectively, $\emptyset$ represents the empty relation and Δ the diagonal relation $\{(a,a) | a \in A\}$. Operation $*$ is Kleene closure, recursively defined, for each $R \in \mathcal{P}(A^2)$, by $R^* = \bigcup_{n \leq \omega} R^n$, where $R^0 = \Delta$ and $R^{n+1} = R^n \circ R$. Finally the residuum is given by $Q \setminus R = \{(x,y) |$ for every z if $(y,z) \in Q$ then $(x,z) \in R\}$. The action lattice is complete.

Example 7 *(**LAN**(Σ)-languages over an alphabet Σ).* Let us consider the action lattice defined by the finite languages on a finite alphabet Σ. Then, for a given finite alphabet Σ, we define the action lattice of languages over Σ as

$$\mathbf{LAN}(\Sigma) = (\mathcal{P}(\Sigma^*), \cup, \cdot, \emptyset, \{\epsilon\}, *, \rightarrow, \cap)$$

where $\cup$ and $\cap$ stand for set union and intersection, $\emptyset$ represents the empty language, ϵ is the empty word, the operation $*$ is the Kleene star defined by $L^* = \bigcup_{n \geq 0} L^n = \{w_1 \cdot \cdots \cdot w_n | w_i \in L, 1 \leq i \leq n\}$ and $L^0 = \{\epsilon\}$ and $L^{n+1} = L \cdot L^n$. The composition $\cdot$ is defined by $L_1 \cdot L_2 = \{w_1 \cdot w_2 | w_1 \in L_1$ and $w_2 \in L_2\}$ and the residuum $\rightarrow$ by $L_1 \rightarrow L_2 = \{v | \forall u (u \in L_1 \Rightarrow u \cdot v \in L_2)\}$. The action lattice is complete.

Example 8 *(**W**$_k$ finite Wajsberg hoops).* We consider now an action lattice endowing the finite *Wajsberg hoops* [4] with a suitable star operation. Hence, for a fix natural $k > 0$ and a generator a, we define the structure

$$\mathbf{W}_k = (W_k, +, ;, 0, 1, *, \rightarrow, \cdot)$$

where $W_k = \{a^0, a^1, \cdots, a^{k-1}\}$, $1 = a^0$ and $0 = a^{k-1}$. Moreover, for any $m, n \leq k - 1$, $a^m + a^n = a^{min\{m,n\}}$, $a^m ; a^n = a^{m+n}$, $(a^m)^* = a^0$, $a^m \rightarrow a^n = a^{max\{n-m,0\}}$ and $a^m \cdot a^n = a^{max\{m,n\}}$.

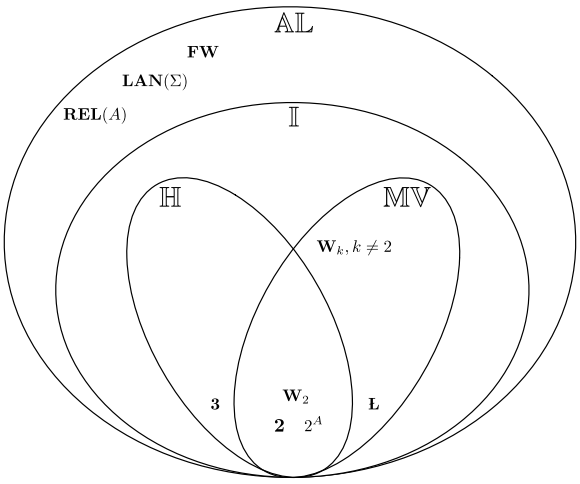


Fig. 2. Examples in the $\mathbb{AL}$ hierarchy.

2.2. $\mathbb{I}$, $\mathbb{H}$ and $\mathbb{MV}$-action lattices

This section introduces three strict classes of action lattices ($\mathbb{AL}$) which become much relevant in the context of the present work. The first class is sound with respect to the positive existential fragment of PDL; the second one is also sound with respect to the remaining positive axioms involving box modalities and tests. The third one captures the standard duality between [_] and $\langle\_\rangle$ modalities. Thus,

Definition 2. Let $\mathbf{A} = (A, +, ;, 0, 1, *, \rightarrow, \cdot)$ be an action lattice. Then $\mathbf{A}$ is said to be

1. A $\mathbb{I}$-*action lattice*, when the identity of the Kleene algebra coincides with the greatest element of the residuated lattice, *i.e.*,
$$1 = \top \tag{46}$$
2. A $\mathbb{H}$-*action lattice*, if it belongs to the class of $\mathbb{I}$-action lattices whose Kleene algebra composition coincides with the infimum of the residuated lattice, *i.e.*, such that for any $a, b \in A$,
$$a; b = a \cdot b \tag{47}$$
3. A $\mathbb{MV}$-*action lattice*, if it is on the class of $\mathbb{I}$-action lattices and satisfies condition
$$(x \rightarrow y) \rightarrow y = x + y \tag{48}$$

The symbols $\mathbb{I}$ $\mathbb{H}$, $\mathbb{MV}$ refer to the residuated lattices from which these properties come from, in particular, integral, Heyting and MV algebras. We can find a succinct presentation of these classes in the survey [19]. For a more complete account of such algebraic structures supporting many-valued logics see books [7] and [14].

With exception of Examples 5, 6, and 7, all the action lattices introduced in the previous section are $\mathbb{I}$-action lattices. This comes from observing that the greatest element of each supporting set, w.r.t. the order induced by $+$, is the identity of ;.

Since their realisations of $\cdot$ and ; coincide, the structures presented in Examples 1, 3, and 2, as well as, the action lattice of the case $k = 2$ in Example 8, are examples of $\mathbb{H}$-action lattices.

Moreover, the $\mathbb{H}$-action lattices of Examples 1 and 3 are examples of $\mathbb{MV}$-action lattices – for **2**, (48) can be easily checked by constructing a truth table and for 2^A, we observe that $(X \rightarrow Y) \rightarrow Y = (X^c \cup Y)^c \cup Y = (X^{cc} \cap Y^c) \cup Y = (X^{cc} \cup Y) \cap (Y^c \cup Y) = (X \cup Y) \cap A = X \cup Y$. The lattice **3** of Example 2 illustrates a $\mathbb{H}$-action lattice that is not a $\mathbb{MV}$-action lattice (since $(u \rightarrow \bot) \rightarrow \bot = \top \neq u + \bot$). The action lattice Ł of Example 4 illustrates a $\mathbb{MV}$-action lattice that is not an $\mathbb{H}$-action lattice. In fact, condition (48) can be verified for Example 4 as follows:

$$
\begin{array}{ll}
 & (x \to y) \to y \\
\Leftrightarrow & \quad \{ \to \text{ interpretation in Ł twice} \} \\
 & min\Big\{1, 1 - min\{1, 1 - x + y\} + y\Big\} \\
\Leftrightarrow & \quad \{ -min\{a, b\} = max\{-a, -b\} \} \\
 & min\Big\{1, 1 + max\{-1, x - 1 - y\} + y\Big\} \\
\Leftrightarrow & \quad \{ max\{a + x, a + y\} = a + max\{x, y\} \} \\
 & min\Big\{1, max\{0, x - y\} + y\Big\}
\end{array}
\qquad
\begin{array}{ll}
\Leftrightarrow & \quad \{ max\{a + x, a + y\} = a + max\{x, y\} \} \\
 & min\Big\{1, max\{y, x\}\Big\} \\
\Leftrightarrow & \quad \{ \text{since } x, y \in [0, 1] \} \\
 & max\{y, x\} \\
\Leftrightarrow & \quad \{ + \text{ interpretation in Ł} \} \\
 & x + y
\end{array}
$$

Finally, we can observe that, for any k, the action lattices of Example 8 are $\mathbb{MV}$-action lattices. Actually, by the definition of W_k, we have that: $(a^m \to a^n) \to a^n = a^{max\{n-m,0\}} \to a^n = a^{max\{n-max\{n-m,0\},0\}} = a^{max\{n+min\{m-n,0\},0\}} = a^{min(m,n)} = a^m + a^n$. The examples discussed above are summarised in Fig. 2.

The following lemma is a very important result for what follows.

Lemma 2. *The following properties hold in any $\mathbb{I}$-lattice:*

1. $(a \to 1) = 1$
2. $(1 \to a) = a$
3. $(\bot \to a) = 1$
4. $(a \to a) = 1$
5. $1 = a; b \Leftrightarrow a = 1 \,\&\, b = 1$
6. $a \leq b \Leftrightarrow (a \to b) = 1$
7. $a = b \Leftrightarrow (a \leftrightarrow b) = 1$

Proof. In order to prove 1., note that rules (46), (6) and (13) entail $a \leq 1 \Leftrightarrow a; 1 \leq 1 \Leftrightarrow 1 \leq (a \to 1)$. By (46), $1 = (a \to 1)$.

In order to prove 2., we have by (6) and (13), $a \leq a \Leftrightarrow 1; a \leq a \Leftrightarrow a \leq (1 \to a)$. Moreover, by (6) and (22), $(1 \to a) = 1; (1 \to a) \leq a$. I.e., $(1 \to a) = a$.

In order to prove 3., we have by (6) and (13), $\bot \leq a \Leftrightarrow \bot; 1 \leq a \Leftrightarrow 1 \leq (\bot \to a)$. Since (46), we have $1 \leq (\bot \to a) \leq 1$, *i.e.*, $(\bot \to a) = 1$.

For 4. we just need to observe, by (6) and (13) $a; 1 \leq a \Leftrightarrow 1 \leq (a \to a)$. By (46), $1 = (a \to a)$.

In order to prove 5., we have by (13) and 2. that $a; b \leq 1 \Leftrightarrow b \leq (a \to 1) = 1$. Since $1 = \top$, $b = 1$. Then, by (6), $a; 1 = 1 \Leftrightarrow a = 1$.

For 6., by (6), (13) and (46), we have $a \leq b \Leftrightarrow a; 1 \leq b \Leftrightarrow 1 \leq a \to b \Leftrightarrow 1 = (a \to b)$.

In order to prove 7. we have

$$
\begin{array}{ll}
 & a = b \\
\Leftrightarrow & \quad \{ = \text{ defn} \} \\
 & a \leq b \,\&\, b \leq a \\
\Leftrightarrow & \quad \{ 6. \} \\
 & 1 = a \to b \,\&\, 1 = b \to a
\end{array}
\qquad
\begin{array}{ll}
\Leftrightarrow & \quad \{ \text{ 5. and ; is functional} \} \\
 & 1; 1 = (a \to b); (b \to a) \\
\Leftrightarrow & \quad \{ \text{by (46) and (6)} \} \\
 & 1 = (a \leftrightarrow b)
\end{array}
\qquad \square
$$

Proposition 1. *Let* **A** *be a $\mathbb{I}$-action lattice. Then, if* **A** *is a $\mathbb{H}$-action lattice, property*

$$(a \to \bot) \cdot a = \bot \tag{49}$$

holds. Moreover, if **A** *is also a $\mathbb{MV}$-action lattice, the property*

$$(a \to \bot) + a = 1 \tag{50}$$

also holds.

Proof. In order to prove (49), we observe that

$$
\begin{array}{ll}
 & a \to \bot \leq a \to \bot \\
\Leftrightarrow & \quad \{ (13) \} \\
 & a; (a \to \bot) \leq \bot
\end{array}
\qquad
\begin{array}{ll}
\Leftrightarrow & \quad \{ (47) + \bot \text{ is the smallest in } \mathbf{A} \} \\
 & a \cdot (a \to \bot) = \bot
\end{array}
$$

The proof of (50) can be done as follows:

$$\begin{array}{lll|lll} & (a \to \bot) + a & & & & \\ = & \{\,(48)\,\} & & = & \{\,(49)\,\} & \\ & ((a \to \bot) \to a) \to a & & & \bot \to a & \square \\ = & \{\,(40)\,\} & & = & \{\,3.\ \text{of Lemma 2} + (46)\,\} & \\ & (a; (a \to \bot)) \to a & & & 1 & \end{array}$$

Note that conditions (49) and (50) actually fail for $\mathbb{MV}$ and $\mathbb{H}$-action lattices, respectively – (49) fails in the $\mathbb{H}$-action lattice **3**, and (50), fails in the $\mathbb{H}$-action lattice Ł. Finally, we note that, as a consequence, every $\mathbb{MV}$-action lattice which is also a $\mathbb{H}$-action lattice, satisfies the entire axiomatics of Boolean algebras.

3. Parametric construction of many-valued dynamic logics

3.1. The method

Once revisited the notion of an action lattice, we are now prepared to introduce the general construction of many-valued dynamic logics. We will therefore introduce its signatures, formulæ, semantics and satisfaction, on top of an arbitrary complete action lattice $\mathbf{A} = (A, +, ;, 0, 1, *, \to, \cdot)$. The logic obtained is denoted by $\mathcal{GDL}(\mathbf{A})$.

Signatures. Signatures of $\mathcal{GDL}(\mathbf{A})$ are pairs (Π, Prop) corresponding to the denotations of atomic computations and propositions, respectively.

Formulæ. A core ingredient of any dynamic logic is its set of programs. Let us denote the set of atomic programs by Π. The *set of* Π*-programs*, denoted by $\mathrm{Prg}(\Pi)$, consists of all expressions generated by

$$\pi \ni \pi_0 \mid \pi ; \pi \mid \pi + \pi \mid \pi^*$$

for $\pi_0 \in \Pi$. Given a signature (Π, Prop), we define the $\mathcal{GDL}(\mathbf{A})$-formulæ for (Π, Prop), denoted by $\mathrm{Fm}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop})$, as the ones generated by grammar

$$\rho \ni \top \mid \bot \mid p \mid \rho \vee \rho \mid \rho \wedge \rho \mid \rho \to \rho \mid \rho \leftrightarrow \rho \mid \langle \pi \rangle \rho \mid [\pi] \rho$$

for $p \in \mathrm{Prop}$ and $\pi \in \mathrm{Prg}(\Pi)$. Note that this corresponds to the *positive* fragment of the propositional dynamic logic.

Semantics. The first step is to introduce the space where the computations of $\mathcal{GDL}(\mathbf{A})$ are to be interpreted. Based on the classic matricial constructions over Kleene algebras (see [8,22]) define

$$\mathbb{M}_n(\mathbf{A}) = (M_n(\mathbf{A}), +, \mathbf{;}, \mathbf{0}, \mathbf{1}, *)$$

as follows:

1. $M_n(\mathbf{A})$ is the space of $(n \times n)$-matrices over $\mathbf{A}$.
2. For any $A, B \in M_n(\mathbf{A})$, define $M = A + B$ by $M_{ij} = A_{ij} + B_{ij}$, $i, j \leq n$.
3. For any $A, B \in M_n(\mathbf{A})$, define $M = A \mathbf{;} B$ by $M_{ij} = \sum_{k=1}^{n} (A_{ik}; B_{kj})$ for any $i, j \leq n$.
4. $\mathbf{1}$ and $\mathbf{0}$ are the $(n \times n)$-matrices defined by $\mathbf{1}_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$ and $\mathbf{0}_{ij} = 0$, for any $i, j \leq n$.
5. For any $M = [a] \in \mathbb{M}_1(\mathbf{A})$, $M^* = [a^*]$;
   for any $M = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \in M_n(\mathbf{A})$, $n > 1$, where A and D are square matrices, define

$$M^* = \left[\begin{array}{c|c} F^* & F^* \mathbf{;} B \mathbf{;} D^* \\ \hline D^*; C; F^* & D^* + (D^* \mathbf{;} C \mathbf{;} F^* \mathbf{;} B \mathbf{;} D^*) \end{array}\right]$$

   where $F = A + B \mathbf{;} D^* \mathbf{;} C$. Note that this construction is recursively defined from the base case (where $n = 2$) where the operations of the base action lattice $\mathbf{A}$ are used.

Finally, a classic result (*e.g.*, [8,22]) establishes that Kleene algebras are closed under formation of matrices.

Theorem 2. *The structure* $\mathbb{M}_n(\mathbf{A}) = (M_n(\mathbf{A}), +, \mathbf{;}, \mathbf{0}, \mathbf{1}, *)$ *defined above is a Kleene algebra.*

$\mathcal{GDL}(\mathbf{A})$-models for a set of propositions Prop and programs Π, denoted by $\mathrm{Mod}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop})$, consists of tuples

$$\mathcal{A} = (W, V, (\mathcal{A}_\pi)_{\pi\in\Pi})$$

where W is a finite set (of states), $V : \mathrm{Prop} \times W \to A$ is a function, and $\mathcal{A}_\pi \in \mathbb{M}_n(\mathbf{A})$, with n standing for the cardinality of W.

The interpretation of programs in these models belongs to the space of the matrices over the Kleene algebra of **A**. Each matrix represents the effect of a program executing from any point of the model. Formally, the interpretation of a program $\pi \in \mathrm{Prg}(\Pi)$ in a model $\mathcal{A} \in \mathrm{Mod}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop})$ is recursively defined, from the set of atomic programs $(\mathcal{A}_\pi)_{\pi\in\Pi}$, as follows:

$$\mathcal{A}_{\pi;\pi'} = \mathcal{A}_\pi \mathbf{;} \mathcal{A}_{\pi'}, \mathcal{A}_{\pi+\pi'} = \mathcal{A}_\pi + \mathcal{A}_{\pi'} \text{ and } \mathcal{A}_{\pi^*} = \mathcal{A}_\pi^*,$$

together with the constants interpretations $\mathcal{A}_1 = \mathbf{1}$ and $\mathcal{A}_0 = \mathbf{0}$. Note that the adoption of the classic matricial constructions on Kleene algebras [8,22], where the operations are defined for $n \times n$ matrices, assumes the finiteness of the state spaces (since n stands for the cardinality of W). Although, sum, composition and union can be easily generalised for the infinite setting, it is not clear if a similar construction exists for the star operation.

Observe that the set of states W supports the index system of the program (adjacency) matrices. In this context, it is important to note that, for example,

$$\mathcal{A}_{\pi;\pi'}(w, w') = \sum_{w''\in W} \big(\mathcal{A}_\pi(w, w''); (\mathcal{A}_{\pi'}(w'', w')\big)$$

should be understood as

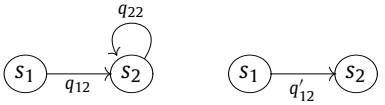
$$(\mathcal{A}_{\pi;\pi'})_{ij} = \sum_{k=1}^{n} \big((\mathcal{A}_\pi)_{ik}; (\mathcal{A}_{\pi'})_{kj}\big)$$

as i and j stands for the adjacency index of w and w', respectively. Actually, the latter characterisation is often used in the sequel.

Example 9 *(Multitude of computational/truth spaces).* Let us fix a complete action lattice $\mathbf{A} = (A, +, ;, 0, 1, *, \to, \cdot)$ and a signature $(\{\pi, \pi'\}, \{p\})$. Now, consider model $\mathcal{A} = (W, V, (\mathcal{A}_\pi)_{\pi\in\Pi})$, with $W = \{s_1, s_2\}$ and the following atomic programs

$$\mathcal{A}_\pi = \begin{bmatrix} \bot & q_{12} \\ \bot & q_{22} \end{bmatrix} \qquad \mathcal{A}_{\pi'} = \begin{bmatrix} \bot & q'_{12} \\ \bot & \bot \end{bmatrix}$$

which can be represented by the following labelled transition systems:



Let $\mathbf{A} = \mathbf{2}$. Taking $q_{12} = q_{22} = q'_{1,2} = \top$ we get the standard adjacency matrices of the graph underlying the transition systems. In this case, we interpret choice $\pi + \pi'$ by

$$\mathcal{A}_{\pi+\pi'} = \mathcal{A}_\pi + \mathcal{A}_{\pi'} = \begin{bmatrix} \bot & \top \\ \bot & \top \end{bmatrix} + \begin{bmatrix} \bot & \top \\ \bot & \bot \end{bmatrix} = \begin{bmatrix} \bot \vee \bot & \top \vee \top \\ \bot \vee \bot & \top \vee \bot \end{bmatrix} = \begin{bmatrix} \bot & \top \\ \bot & \top \end{bmatrix}$$

For the interpretation of the π closure, we have

$$\mathcal{A}_{\pi^*} = (\mathcal{A}_\pi)^* \begin{bmatrix} \bot & \top \\ \bot & \top \end{bmatrix}^* = \begin{bmatrix} f^* & f^* \wedge \top \wedge \top^* \\ \top^* \wedge \bot \wedge \bot^* & \top^* \vee (\top^* \wedge \bot \wedge \top \wedge \top) \end{bmatrix}$$

where $f = \bot \vee (\top \wedge \top^* \wedge \bot) = \bot$; hence $\mathcal{A}_{\pi*} = \begin{bmatrix} \top & \top \\ \bot & \top \end{bmatrix}$.

Taking the same matrix for $\mathbf{A} = \mathbf{3}$ and considering $q_{12} = q_{22} = \top$ and $q'_{12} = u$, we have

$$\mathcal{A}_{\pi';\pi} = \begin{bmatrix} \bot & u \\ \bot & \bot \end{bmatrix} \mathbf{;} \begin{bmatrix} \bot & \top \\ \bot & \top \end{bmatrix} = \begin{bmatrix} (\bot \wedge \bot) \vee (u \wedge \bot) & (\bot \wedge \top) \vee (u \wedge \top) \\ (\bot \wedge \bot) \vee (\bot \wedge \bot) & (\bot \wedge \top) \vee (\bot \wedge \top) \end{bmatrix} = \begin{bmatrix} \bot & u \\ \bot & \bot \end{bmatrix}.$$

As expected, the unknown factor affecting transition $s_1 \to s_2$ in $\mathcal{A}'_\pi$, is propagated to transition $s_1 \to s_2$ in $\mathcal{A}_{\pi';\pi}$. If a continuous space is required to define the "unknown" metric, the Łukasiewicz arithmetic lattice **Ł** would be a suitable

option. Consider, for instance, $q_{12} = a$, $q_{22} = b$ and $q'_{12} = c$ for some $a, b, c \in [0, 1]$. In this case we may, for example, compute $\mathcal{A}_{\pi+\pi'}$ as follows:

$$\mathcal{A}_{\pi+\pi'} = \begin{bmatrix} 0 & a \\ 0 & b \end{bmatrix} + \begin{bmatrix} 0 & c \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} max\{0,0\} & max\{a,c\} \\ max\{0,0\} & max\{b,0\} \end{bmatrix} = \begin{bmatrix} 0 & max\{a,c\} \\ 0 & b \end{bmatrix}.$$

The reader may check that

$$\mathcal{A}_{\pi^*} = \begin{bmatrix} 0 & a \\ 0 & b \end{bmatrix}^* = \begin{bmatrix} f^* & max\{0, max\{0, f^* + a - 1\} + b^* - 1\} \\ (b \odot 0) \odot f^* & max\{f^*, \cdots\} \end{bmatrix} = \begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}.$$

Let now illustrate the program interpretation in other less standard spaces of computation and truth. Taking the action lattice **REL** and considering $q_{12} = R$, $q_{22} = A \times A$ and $q'_{12} = R^*$ for some $R \in \mathcal{P}(A \times A)$. In this case we may, for example, compute choice $\pi + \pi'$, making

$$\mathcal{A}_{\pi+\pi'} = \begin{bmatrix} \emptyset & R \\ \emptyset & A^2 \end{bmatrix} + \begin{bmatrix} \emptyset & R^* \\ \emptyset & \emptyset \end{bmatrix} = \begin{bmatrix} \emptyset & R \cup R^* \\ \emptyset & A^2 \end{bmatrix}$$

As expected, in this case the computation transitions are weighted by relations. Choosing now action lattice **LAN**, these transitions are weighted by languages. For instance, by considering $q_{12} = \{\epsilon\}$, $q_{22} = \{a\}$, we may compute the program π^* by taking $f = \emptyset + \{\epsilon\}; \{a\}^*; \emptyset = \emptyset$, hence $f^* = \{\epsilon\}$, and therefore

$$\mathcal{A}_{\pi^*} = \begin{bmatrix} \{\epsilon\} & \{\epsilon\} \cap \{\epsilon\} \cap \{a\}^* \\ \{a\}^* \cap \emptyset \cap f^* & \{a\}^* \cup \{a\}^* \cap \emptyset \cap \{\epsilon\} \cap \{\epsilon\} \cap \{a\}^* \end{bmatrix} = \begin{bmatrix} \{\epsilon\} & \{a\}^* \\ \emptyset & \{a\}^* \end{bmatrix}$$

or, resorting got the notations of regular expressions, $\mathcal{A}_{\pi^*} = \begin{bmatrix} \epsilon & a^* \\ \emptyset & a^* \end{bmatrix}$. We will adopt this notation in the sequel. Examples of interpretations of programs in action lattice $\mathbb{N}^+_{\perp\top}$ can be found in [28,29].

Satisfaction. Finally, let us define the (graded) satisfaction relation. This kind of satisfaction relation was already considered, in the context of many-valued modal logics by M. Fitting in [11,12] and, more recently, by F. Bou in [6].

As mentioned above, the carrier of **A** corresponds to the space of truth degrees for $\mathcal{GDL}(\mathbf{A})$. Hence, the graded satisfaction relation for a model $\mathcal{A} \in \mathrm{Mod}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop})$, with **A** complete, consists of a function

$$\models : W \times \mathrm{Fm}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop}) \to A$$

recursively defined as follows:

- $(w \models \top) = \top$
- $(w \models \bot) = \bot$
- $(w \models p) = V(p, w)$, for any $p \in \mathrm{Prop}$
- $(w \models \rho \wedge \rho') = (w \models \rho) \cdot (w \models \rho')$
- $(w \models \rho \vee \rho') = (w \models \rho) + (w \models \rho')$
- $(w \models \rho \to \rho') = (w \models \rho) \to (w \models \rho')$
- $(w \models \rho \leftrightarrow \rho') = (w \models \rho \to \rho'); (w \models \rho' \to \rho)$
- $(w \models \langle\pi\rangle\rho) = \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); (w' \models \rho)\right)$
- $(w \models [\pi]\rho) = \prod_{w' \in W} \left(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\right)$

We say that ρ is *valid* when, for any model $\mathcal{A}$, and for each state $w \in W$, $(w \models \rho) = \top$. As stated above, we use, in the semantics of formulæ with equivalences, the adjunct of the implication ;, instead of (standard) conjunction $\cdot$. Actually, the operation ; has a double role, acting as composition and as "strong conjunction" with respect to the underlying Kleene algebra and to the residuated lattice. The following result will be useful in the sequel:

Lemma 3. *Let $\mathcal{A}$ be a complete $\mathbb{I}$-action lattice. Then*

- $(w \models \rho \to \rho') = \top$ *iff* $(w \models \rho) \leq (w \models \rho')$
- $(w \models \rho \leftrightarrow \rho') = \top$ *iff* $(w \models \rho) = (w \models \rho')$

Proof. Since $\mathcal{A}$ is a $\mathbb{I}$-action lattice we have $\top = 1$ and hence, we conclude both equivalences by clauses 5. and 6. of Lemma 2, respectively. □

Example 10. In order to illustrate the versatility and generality of the method, let us consider the evaluation of the very simple sentence $\langle \pi^* \rangle p$ in three of the dynamic logics constructed in the examples above. Concretely, let us evaluate $\langle \pi^* \rangle p$ in state s_1. For this we calculate

$$(s_1 \models \langle \pi^* \rangle p) = \sum_{w' \in W} \big(\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p) \big)$$

Starting with $\mathcal{GDL}(\mathbf{2})$, let us assume $V(p, s_1) = \bot$ and $V(p, s_2) = \top$. In this case, as expected

$$\begin{aligned}
(s_1 \models \langle \pi^* \rangle p) &= \sum_{w' \in W} \big(\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p) \big) \\
&= \big(\mathcal{A}_{\pi^*}(s_1, s_1) \wedge (s_1 \models p) \big) \vee \big(\mathcal{A}_{\pi^*}(s_1, s_2) \wedge (s_2 \models p) \big) \\
&= (\top \wedge V(p, s_1)) \vee (\top \wedge V(p, s_2)) \\
&= (\top \wedge \bot) \vee (\top \wedge \top) \\
&= \top
\end{aligned}$$

This means that p can be achieved from s_1 through π^*.

Considering the $\mathcal{GDL}(Ł)$ and assuming $V(s_1, p) = 0$ and $V(s_2, p) = 1$, we calculate

$$\begin{aligned}
(s_1 \models \langle \pi^* \rangle p) &= \sum_{w' \in W} \big(\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p) \big) \\
&= max\big\{ \\
&\quad max\{0, (s_1 \models p) + \mathcal{A}_{\pi^*}(s_1, s_1) - 1\}, \\
&\quad max\{0, (s_2 \models p) + \mathcal{A}_{\pi^*}(s_1, s_2) - 1\} \\
&\quad \big\} \\
&= max\{0, a\} \\
&= a
\end{aligned}$$

Therefore, we can assure, with a degree of certainty a, that p is achieved from s_1 through π^*.

More 'exotic' examples can be used in this illustration. For instance, making the same evaluation in the logic $\mathcal{GDL}(\mathbf{LAN})$ by assuming $V(s_1, p) = \epsilon$ and $V(s_2, p) = a; a$, we get,

$$\begin{aligned}
(s_1 \models \langle \pi^* \rangle p) &= \sum_{w' \in W} \big(\mathcal{A}_{\pi^*}(s_1, w'); (w' \models p) \big) \\
&= \epsilon; \epsilon + a^*; aa + \emptyset; \epsilon + a^*; a; a \\
&= \epsilon + a^* + a^* \\
&= a^*
\end{aligned}$$

3.2. How dynamic are dynamisations?

Having introduced a generic method for generating dynamic logics, this section establishes that the resulting logics behave, in fact, as expected. In the sequel by dynamic logic we understand the classical, propositional version [41,17].

3.2.1. $\langle \_ \rangle$-modalities

This section studies axioms involving $\langle \_ \rangle$-modalities.

Lemma 4. *Let* **A** *be a complete* $\mathbb{I}$*-action lattice. The following are valid formulæ in any* $\mathcal{GDL}(\mathbf{A})$*:*

(4.1) $\langle \pi \rangle(\rho \vee \rho') \leftrightarrow \langle \pi \rangle \rho \vee \langle \pi \rangle \rho'$
(4.2) $\langle \pi \rangle(\rho \wedge \rho') \rightarrow \langle \pi \rangle \rho \wedge \langle \pi \rangle \rho'$

Proof. To establish the validity of **(4.1)** we reason

$$
\begin{array}{ll}
 & (w \models \langle\pi\rangle(\rho \vee \rho')) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho \vee \rho')\big) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \sum_{w' \in W} \big((\mathcal{A}_\pi(w, w'); \\
 & \big((w' \models \rho) + (w' \models \rho')\big)\big) \\
= & \quad \{ (7) \} \\
 & \sum_{w' \in W} \big((\mathcal{A}_\pi(w, w'); (w' \models \rho) + \\
 & (\mathcal{A}_\pi(w, w'); (w' \models \rho'))\big)
\end{array}
\qquad
\begin{array}{ll}
= & \quad \{ \text{ by (1) and (2)} \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho)\big) + \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho')\big) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho) + (w \models \langle\pi\rangle\rho) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho \vee \langle\pi\rangle\rho)
\end{array}
$$

Therefore, by Lemma 3, $\langle\pi\rangle(\rho \vee \rho') \leftrightarrow \langle\pi\rangle\rho \vee \langle\pi\rangle\rho$ is valid.

The case of **(4.2)** follows similarly,

$$
\begin{array}{ll}
 & (w \models \langle\pi\rangle(\rho \wedge \rho')) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho \wedge \rho')\big) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); \\
 & \big((w' \models \rho) \cdot (w' \models \rho')\big)\big) \\
\leq & \quad \{ \text{ by (36) and (33)} \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho) \cdot \\
 & (\mathcal{A}_\pi(w, w'); (w' \models \rho'))\big)
\end{array}
\qquad
\begin{array}{ll}
\leq & \quad \{ \text{ by (45) } \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho)\big) \cdot \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho')\big) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho) \cdot (w \models \langle\pi\rangle\rho') \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho \wedge \langle\pi\rangle\rho')
\end{array}
$$

Therefore, by Lemma 3, $\langle\pi\rangle(\rho \wedge \rho') \rightarrow \langle\pi\rangle\rho \wedge \langle\pi\rangle\rho'$ is valid. □

Lemma 5. *Let* **A** *be a complete* $\mathbb{I}$*-action lattice. The following are valid formulæ in* $\mathcal{GDL}(\mathbf{A})$*:*

(5.1) $\langle\pi + \pi'\rangle\rho \leftrightarrow \langle\pi\rangle\rho \vee \langle\pi\rangle\rho$
(5.2) $\langle\pi ; \pi'\rangle\rho \leftrightarrow \langle\pi\rangle\langle\pi'\rangle\rho$
(5.3) $\langle\pi\rangle\bot \leftrightarrow \bot$

Proof. The validity of **(5.1)** is established as follows:

$$
\begin{array}{ll}
 & (w \models \langle\pi + \pi'\rangle\rho) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_{\pi+\pi'}(w, w'); (w' \models \rho)\big) \\
= & \quad \{ \text{ program's interpretation} \} \\
 & \sum_{w' \in W} \big((\mathcal{A}_\pi(w, w') + \mathcal{A}_{\pi'}(w, w')); \\
 & (w' \models \rho)\big) \\
= & \quad \{ \text{ by (8)} \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho) + \\
 & \mathcal{A}_{\pi'}(w, w'); (w' \models \rho)\big)
\end{array}
\qquad
\begin{array}{ll}
= & \quad \{ \text{ by (1) and (2)} \} \\
 & \sum_{w' \in W} \big(\mathcal{A}_\pi(w, w'); (w' \models \rho)\big) + \\
 & \sum_{w' \in W} \big(\mathcal{A}_{\pi'}(w, w'); (w' \models \rho)\big) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho) + (w \models \langle\pi'\rangle\rho) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho \vee \langle\pi'\rangle\rho)
\end{array}
$$

Therefore, by Lemma 3, $\langle\pi + \pi'\rangle\rho \leftrightarrow \langle\pi\rangle\rho \vee \langle\pi'\rangle\rho$ is valid.

To establish the validity of **(5.2)** we observe that

$$
\begin{array}{ll}
 & (w \models \langle\pi\rangle\langle\pi'\rangle\rho) \\
= & \quad \{ \text{defn of } \models \} \\
 & \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); (w \models \langle\pi'\rangle\rho) \right) \\
= & \quad \{ \text{defn of } \models \} \\
 & \sum_{w' \in W} \Big(\mathcal{A}_\pi(w, w'); \\
 & \sum_{w'' \in W} \left(\mathcal{A}_{\pi'}(w', w''); (w'' \models \rho) \right) \Big) \\
= & \quad \{ \text{by (7)} \} \\
 & \sum_{w' \in W} \Big(\sum_{w'' \in W} (\\
 & \mathcal{A}_\pi(w, w'); (\mathcal{A}_{\pi'}(w', w''); (w'' \models \rho)) \Big) \\
= & \quad \{ \text{by (1) and (2)} \}
\end{array}
\qquad
\begin{array}{ll}
 & \sum_{w'' \in W} \Big(\sum_{w' \in W} (\\
 & \mathcal{A}_\pi(w, w'); \mathcal{A}_{\pi'}(w', w'')); (w'' \models \rho) \Big) \\
= & \quad \{ (w'' \models \rho) \text{ is independent of } w' \} \\
 & \sum_{w'' \in W} \Big(\sum_{w' \in W} (\\
 & \mathcal{A}_\pi(w, w'); \mathcal{A}_{\pi'}(w', w'')); (w'' \models \rho) \Big) \\
= & \quad \{ \text{defn of composition} \} \\
 & \sum_{w'' \in W} \left(\mathcal{A}_{\pi;\pi'}(w, w''); (w'' \models \rho) \right) \\
= & \quad \{ \text{defn of } \models \} \\
 & (w \models \langle\pi;\pi'\rangle\rho)
\end{array}
$$

Therefore, by Lemma 3, $\langle\pi\rangle\langle\pi'\rangle\rho \leftrightarrow \langle\pi;\pi'\rangle\rho$ is valid.

The validity of **(5.3)** is proved by observing that

$$
\begin{array}{ll}
 & (w \models \langle\pi\rangle\bot) \\
= & \quad \{ \text{defn of } \models \} \\
 & \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); (w' \models \bot) \right) \\
= & \quad \{ \text{defn of satisfaction} \} \\
 & \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); \bot \right)
\end{array}
\qquad
\begin{array}{ll}
= & \quad \{ \text{by (9)} \} \\
 & \sum_{w' \in W} (\bot) \\
= & \quad \{ \text{by (4)} \} \\
 & \bot
\end{array}
$$

Therefore, by Lemma 3, $\langle\pi\rangle\bot \leftrightarrow \bot$ is valid. □

Lemma 6. *Let* **A** *be a complete* $\mathbb{I}$*-action lattice. The following are valid formulæ in* $\mathcal{GDL}(\mathbf{A})$*:*

(6.1) $\langle\pi\rangle\rho \rightarrow \langle\pi^*\rangle\rho$
(6.2) $\langle\pi^*\rangle\rho \leftrightarrow \langle\pi^*;\pi^*\rangle\rho$
(6.3) $\langle\pi^*\rangle\rho \leftrightarrow \langle\pi^{**}\rangle\rho$
(6.4) $\langle\pi^*\rangle\rho \leftrightarrow \rho \vee \langle\pi\rangle\langle\pi^*\rangle\rho$

Proof. By Theorem 1 and (23)–(26), have:

$$\mathcal{A}_\pi(w, w') \leq \mathcal{A}_{\pi^*}(w, w') \tag{51}$$

$$\mathcal{A}_{\pi^*}(w, w') = \mathcal{A}_{\pi^{**}}(w, w') \tag{52}$$

$$\mathcal{A}_{\pi^*}(w, w') = \mathcal{A}_{\pi^*;\pi^*}(w, w') \tag{53}$$

$$\mathcal{A}_{1+\pi;\pi^*}(w, w') = \mathcal{A}_{\pi^*}(w, w') \tag{54}$$

Hence, to verify **(6.1)**, we reason

$$
\begin{array}{ll}
 & \text{for any } w' \in W \\
 & \mathcal{A}_\pi(w, w') \leq \mathcal{A}_{\pi^*}(w, w') \\
\Rightarrow & \quad \{ \text{by (32)} \} \\
 & \text{for any } w' \in W \\
 & \mathcal{A}_\pi(w, w'); (w' \models \rho) \leq \\
 & \mathcal{A}_{\pi^*}(w, w'); (w' \models \rho)
\end{array}
\qquad
\begin{array}{ll}
\Rightarrow & \quad \{ \text{by (33)} \} \\
 & \sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); (w' \models \rho) \right) \leq \\
 & \sum_{w' \in W} \left(\mathcal{A}_{\pi^*}(w, w'); (w' \models \rho) \right) \\
\Leftrightarrow & \quad \{ \text{defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho) \leq (w \models \langle\pi^*\rangle\rho)
\end{array}
$$

Since by (51), $\mathcal{A}_\pi(w, w') \leq \mathcal{A}_{\pi^*}(w, w')$ holds for any $w, w' \in W$, we conclude $(w \models \langle\pi\rangle\rho) \leq (w \models \langle\pi^*\rangle\rho)$. Hence, by Lemma 3, $\langle\pi\rangle\rho \rightarrow \langle\pi^*\rangle\rho$ is valid. The remaining of the first two proofs follows exactly the same steps but starting from (52) and (53).

For **(6.4)**, we have

$$\mathcal{A}_{1+\pi;\pi^*}(w,w') = \mathcal{A}_{\pi^*}(w,w') \text{ for any } w, w' \in W$$

$\Leftrightarrow$ { program interpretation}

$$\mathcal{A}_1(w,w') + \mathcal{A}_{\pi;\pi^*}(w,w') = \mathcal{A}_{\pi^*}(w,w') \text{ for any } w' \in W$$

$\Leftrightarrow$ $\{ a = b \Rightarrow a;c = b;c \}$

$$\big(\mathcal{A}_1(w,w') + \mathcal{A}_{\pi;\pi^*}(w,w')\big);(w' \models \rho) = \mathcal{A}_{\pi^*}(w,w');(w' \models \rho)$$
$$\text{for any } w' \in W$$

$\Leftrightarrow$ { (8)}

$$\mathcal{A}_1(w,w');(w' \models \rho) + \mathcal{A}_{\pi;\pi^*}(w,w');(w' \models \rho) = \mathcal{A}_{\pi^*}(w,w');(w' \models \rho)$$
$$\text{for any } w' \in W$$

$\Leftrightarrow$ $\{ a_i = b_i, i \in I \Rightarrow \sum_{i\in I} a_i = \sum_{i\in I} a_i \}$

$$\sum_{w'\in W} \big(\mathcal{A}_1(w,w');(w' \models \rho) + \mathcal{A}_{\pi;\pi^*}(w,w');(w' \models \rho)\big) =$$
$$\sum_{w'\in W} \big(\mathcal{A}_{\pi^*}(w,w');(w' \models \rho)\big)$$

$\Leftrightarrow$ { by (1) and (2)}

$$\sum_{w'\in W} \big(\mathcal{A}_1(w,w');(w' \models \rho)\big) + \sum_{w'\in W} \big(\mathcal{A}_{\pi;\pi^*}(w,w');(w' \models \rho)\big) =$$
$$\sum_{w'\in W} \big(\mathcal{A}_{\pi^*}(w,w');(w' \models \rho)\big)$$

$\Leftrightarrow$ $\{ \sum_{w'\in W} \big(\mathcal{A}_1(w,w');(w' \models \rho) = (w \models \rho)\big)$ and program interpretation$\}$

$$(w \models \rho) + (w \models \langle \pi;\pi^* \rangle \rho) = (w \models \langle \pi^* \rangle \rho)$$

$\Leftrightarrow$ { **(6.2)**}

$$(w \models \rho) + (w \models \langle \pi \rangle \langle \pi^* \rangle \rho) = (w \models \langle \pi^* \rangle \rho)$$

$\Leftrightarrow$ { defn of $\models$}

$$(w \models \rho \vee \langle \pi \rangle \langle \pi^* \rangle \rho) = (w \models \langle \pi^* \rangle \rho)$$

Therefore, by Lemma 3, $\langle \pi^* \rangle \rho \leftrightarrow \rho \vee \langle \pi \rangle \langle \pi^* \rangle \rho$ holds. □

3.2.2. [_]*-modalities*

We shall now consider the validity of axioms involving [_]-modalities.

Lemma 7. *Let $\mathcal{A}$ be a complete $\mathbb{I}$-action lattice. The following are valid formulæ in $\mathcal{GDL}(\mathbf{A})$:*

(7.1) $[\pi + \pi']\rho \leftrightarrow [\pi]\rho \wedge [\pi']\rho$
(7.2) $[\pi](\rho \wedge \rho') \leftrightarrow [\pi]\rho \wedge [\pi]\rho'$

Proof. Proof of **(7.1)**:

$$
\begin{array}{ll}
 & (w \models [\pi + \pi']\rho) \\
= & \quad \{ \models \text{defn} \} \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi+\pi'}(w, w') \to (w' \models \rho)\big) \\
= & \quad \{ \text{program interpretation} \} \\
 & \prod_{w' \in W} \Big(\big(\mathcal{A}_{\pi}(w, w') + \mathcal{A}_{\pi'}(w, w')\big) \\
 & \to (w' \models \rho)\Big) \\
= & \quad \{ (28) \} \\
 & \prod_{w' \in W} \Big(\big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho)\big) \cdot \\
 & \big(\mathcal{A}_{\pi'}(w, w') \to (w' \models \rho)\big)\Big)
\end{array}
\qquad
\begin{array}{ll}
= & \quad \{ (17) \} \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho)\big) \cdot \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi'}(w, w') \to (w' \models \rho)\big) \\
= & \quad \{ \text{defn of } \models \} \\
 & (w \models [\pi]\rho) \cdot (w \models [\pi']\rho) \\
= & \quad \{ \text{defn of } \models \} \\
 & (w \models [\pi]\rho \wedge [\pi']\rho)
\end{array}
$$

Therefore, by Lemma 3, we have that $[\pi + \pi']\rho \leftrightarrow [\pi]\rho \wedge [\pi']\rho$ is valid.

Proof of **(7.2)**:

$$
\begin{array}{ll}
 & (w \models [\pi](\rho \wedge \rho')) \\
= & \quad \{ \text{defn of } \models \} \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho \wedge \rho')\big) \\
= & \quad \{ \text{defn of } \models \} \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi}(w, w') \to \\
 & ((w' \models \rho) \cdot (w' \models \rho'))\big) \\
= & \quad \{ (27) \} \\
 & \prod_{w' \in W} \Big(\big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho)\big) \cdot \\
 & \big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho')\big)\Big)
\end{array}
\qquad
\begin{array}{ll}
= & \quad \{ (17) \} \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho)\big) \cdot \\
 & \prod_{w' \in W} \big(\mathcal{A}_{\pi}(w, w') \to (w' \models \rho')\big) \\
= & \quad \{ \text{defn of } \models \} \\
 & (w \models [\pi]\rho) \cdot (w \models [\pi]\rho') \\
= & \quad \{ \text{defn of } \models \} \\
 & (w \models [\pi]\rho \wedge [\pi]\rho')
\end{array}
$$

Therefore, by Lemma 3, $[\pi](\rho \wedge \rho') \leftrightarrow [\pi]\rho \wedge [\pi]\rho'$ holds. □

Lemma 8. *Let* **A** *be a complete* $\mathbb{I}$*-action lattice satisfying*

$$a \to (b \to c) = (a; b) \to c \tag{55}$$

Then property

$$[\pi; \pi']\rho \leftrightarrow [\pi][\pi']\rho \tag{56}$$

is valid in $\mathcal{GDL}(\mathbf{A})$.

Proof.

$$
\begin{array}{ll}
 & (w \models [\pi; \pi']\rho) \\
= & \quad \{ \text{defn of } \models \} \\
 & \displaystyle\prod_{w' \in W} \big(\mathcal{A}_{\pi;\pi'}(w, w') \to (w' \models \rho)\big) \\
= & \quad \{ \text{program interpretation} \} \\
 & \displaystyle\prod_{w' \in W} \Big(\big(\sum_{w'' \in W} \big(\mathcal{A}_{\pi}(w, w''); \mathcal{A}_{\pi'}(w'', w')\big)\big) \to (w' \models \rho)\Big) \\
= & \quad \{ \text{by } (28) \} \\
 & \displaystyle\prod_{w' \in W} \Big(\prod_{w'' \in W} \big(\mathcal{A}_{\pi}(w, w''); \mathcal{A}_{\pi'}(w'', w') \to (w' \models \rho)\big)\Big) \\
= & \quad \{ \text{by } (55) \}
\end{array}
$$

$$
\begin{aligned}
&\prod_{w'\in W}\Big(\prod_{w''\in W}\big(\mathcal{A}_\pi(w,w'')\to(\mathcal{A}_{\pi'}(w'',w')\to(w'\models\rho))\big)\Big)\\
=\quad&\{\text{ by (17) and (18)}\}\\
&\prod_{w''\in W}\Big(\prod_{w'\in W}\big(\mathcal{A}_\pi(w,w'')\to(\mathcal{A}_{\pi'}(w'',w')\to(w'\models\rho))\big)\Big)\\
=\quad&\{\text{ by (27)}\}\\
&\prod_{w''\in W}\Big(\mathcal{A}_\pi(w,w'')\to(\prod_{w'\in W}\{\mathcal{A}_{\pi'}(w'',w')\to(w'\models\rho)))\Big)\\
=\quad&\{\text{ defn of }\models\}\\
&\prod_{w''\in W}\big(\mathcal{A}_\pi(w,w'')\to(w''\models[\pi']\rho)\big)\\
=\quad&\{\text{ defn of }\models\}\\
&(w\models[\pi][\pi']\rho)\qquad\square
\end{aligned}
$$

Observe that condition (55) is, in fact, necessary in Lemma 8. We can illustrate this with the following counter-example: Let us consider the following $\mathbb{I}$-lattice:

$\vee$	0	a	b	1
0	0	a	b	1
a	a	a	b	1
b	b	b	b	1
1	1	1	1	1

$\wedge$	0	a	b	1
0	0	0	0	0
a	0	a	a	a
b	0	a	b	b
1	0	a	b	1

$\to$	0	a	b	1
0	1	1	1	1
a	b	1	1	1
b	0	a	1	1
1	0	a	b	1

;	0	a	b	1
0	0	0	0	0
a	0	0	0	a
b	0	a	b	b
1	0	a	b	1

$*$	
0	1
a	1
b	1
1	1

Note that (55) fails since $a\to(b\to 0)=b\neq 1=(a;b)\to 0$. For this $\mathbb{I}$-lattice, let us consider a model $\mathcal{A}$ with $W=\{\bullet\}$, two programs π and π' such that $\mathcal{A}_\pi(\bullet,\bullet)=a$ and $\mathcal{A}_{\pi'}(\bullet,\bullet)=b$ and a proposition p such that $V(\bullet,p)=0$. Hence,

$$
\begin{aligned}
(\bullet\models[\pi;\pi']p)&=\prod_{w\in W}\big(\mathcal{A}_{\pi;\pi'}(\bullet,w)\to(w\models p)\big)\\
&=\mathcal{A}_{\pi;\pi'}(\bullet,\bullet)\to(\bullet\models p)\\
&=(\mathcal{A}_\pi(\bullet,\bullet);\mathcal{A}_{\pi'}(\bullet,\bullet))\to 0\\
&=(a;b)\to 0\\
&=0\to 0\\
&=1
\end{aligned}
$$

On the other hand:

$$
\begin{aligned}
(\bullet\models[\pi][\pi']p)&=\prod_{w\in W}\big(\mathcal{A}_\pi(\bullet,w)\to(w\models[\pi']p)\big)\\
&=\mathcal{A}_\pi(\bullet,\bullet)\to(\bullet\models[\pi']p)\\
&=\mathcal{A}_\pi(\bullet,\bullet)\to(\prod_{w\in W}\big(\mathcal{A}_{\pi'}(\bullet,w)\to(w\models p)\big)\\
&=\mathcal{A}_\pi(\bullet,\bullet)\to(\mathcal{A}_{\pi'}(\bullet,\bullet)\to(\bullet\models p))\\
&=a\to(b\to 0)\\
&=a\to 0\\
&=b
\end{aligned}
$$

Lemma 9. *Let* **A** *be a complete* ;*-commutative* $\mathbb{I}$*-action lattice satisfying*

$$a;a=a \tag{57}$$

Then property

$$[\pi](\rho \to \rho') \to ([\pi]\rho \to [\pi]\rho') \tag{58}$$

is valid in $\mathcal{GDL}(\mathbf{A})$.

Proof. In order to prove (58) observe that $(w \models [\pi](\rho \to \rho')); (w \models [\pi]\rho) = \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to ((w' \models \rho) \to (w' \models \rho'))\big); \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\big)$. By (42) we have: $\prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to ((w' \models \rho) \to (w' \models \rho'))\big); \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\big) \leq \big(\mathcal{A}_\pi(w, w') \to ((w' \models \rho) \to (w' \models \rho'))\big); \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\big)$, for any $w' \in W$.

Moreover, we have for any $w' \in W$,

$$\begin{array}{ll} & \big(\mathcal{A}_\pi(w, w') \to ((w' \models \rho) \to (w' \models \rho'))\big); \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\big) \\ \leq & \quad \{\text{ by hypothesis and (44)}\} \\ & \mathcal{A}_\pi(w, w') \to (w' \models \rho') \end{array}$$

Then, by (41) we have that $\prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to ((w' \models \rho) \to (w' \models \rho'))\big); \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho)\big) \leq \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to (w' \models \rho')\big) = (w \models [\pi]\rho')$.

Moreover,

$$\begin{array}{ll} & (w \models [\pi](\rho \to \rho'); \\ & (w \models [\pi]\rho) \leq (w \models [\pi]\rho') \\ \Leftrightarrow & \quad \{\text{ ;-commutative}\} \\ & (w \models [\pi]\rho); \\ & (w \models [\pi](\rho \to \rho') \leq (w \models [\pi]\rho') \\ \Leftrightarrow & \quad \{\text{ by (13)}\} \end{array} \qquad \begin{array}{ll} & (w \models [\pi](\rho \to \rho') \leq \\ & (w \models [\pi]\rho) \to (w \models [\pi]\rho') \\ \Leftrightarrow & \quad \{\text{ defn of } \models\} \\ & (w \models [\pi](\rho \to \rho') \leq \\ & (w \models [\pi]\rho \to [\pi]\rho') \end{array}$$

Therefore, by Lemma 3, $[\pi](\rho \to \rho') \to ([\pi]\rho \to [\pi]\rho')$. □

Corollary 1. *Suppose that* **A** *is a complete* $\mathbb{H}$*-action lattice. Then, axioms (56) and (58) are valid in* $\mathcal{GDL}(\mathbf{A})$.

Proof. In order to prove the validity of (56) note that (47) entails the commutativity of ;. Hence, (55) can be derived from (40) (by replacing, in the right side of (55), $b; a$ to $a; b$). In order to prove (58) observe that idempotency (57) is also a direct consequence of (47) and (19). □

Lemma 10. *Let* **A** *be a complete* $\mathbb{H}$*-action lattice. Then*

(10.1) $\rho \wedge [\pi][\pi^*]\rho \leftrightarrow [\pi^*]\rho$
(10.2) $[\pi^*](\rho \to [\pi]\rho) \to (\rho \to [\pi^*]\rho)$

are valid in $\mathcal{GDL}(\mathbf{A})$.

Proof. To prove **(10.1)**, we reason

$$\begin{array}{ll} & \mathcal{A}_{1+\pi;\pi^*}(w, w') = \mathcal{A}_{\pi^*}(w, w') \text{ for any } w, w' \in W \\ \Leftrightarrow & \quad \{\text{ since } a = b \Rightarrow (a \to c) = (b \to c)\} \\ & (\mathcal{A}_{1+\pi;\pi^*}(w, w') \to (w' \models \rho)) = (\mathcal{A}_{\pi^*}(w, w') \to (w' \models \rho)) \text{ for any } w' \in W \\ \Leftrightarrow & \quad \{\, a_i = b_i, i \in I \Rightarrow \textstyle\prod_{i \in I} a_i = \prod_{i \in I} b_i\} \\ & \displaystyle\prod_{w' \in W} (\mathcal{A}_{1+\pi;\pi^*}(w, w') \to (w' \models \rho)) = \prod_{w' \in W} (\mathcal{A}_{\pi^*}(w, w') \to (w' \models \rho)) \\ \Leftrightarrow & \quad \{\text{ program interpretation}\} \\ & \displaystyle\prod_{w' \in W} \big((\mathcal{A}_1(w, w') + \mathcal{A}_{\pi;\pi^*}(w, w')) \to (w' \models \rho)\big) = \prod_{w' \in W} \big(\mathcal{A}_{\pi^*}(w, w') \to (w' \models \rho)\big) \\ \Leftrightarrow & \quad \{\text{ by (28)}\} \end{array}$$

$$\prod_{w'\in W}\big((\mathcal{A}_1(w,w')\to(w'\models\rho)\big)\cdot(\mathcal{A}_{\pi;\pi^*}(w,w')\to(w'\models\rho)))$$
$$=\prod_{w'\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to(w'\models\rho)\big)$$

$\Leftrightarrow$ { by (17) and (18)}

$$\prod_{w'\in W}((\mathcal{A}_1(w,w')\to(w'\models\rho))\cdot\prod_{w'\in W}((\mathcal{A}_{\pi;\pi^*}(w,w')\to(w'\models\rho)))$$
$$=\prod_{w'\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to(w'\models\rho)\big)$$

$\Leftrightarrow$ { step $\star$}

$$(w\models\rho)\cdot\prod_{w'\in W}\big(\mathcal{A}_{\pi;\pi^*}(w,w')\to(w'\models\rho)\big))=\prod_{w'\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to(w'\models\rho)\big)$$

$\Leftrightarrow$ { $\models$ defn}

$$(w\models\rho)\cdot(w\models[\pi;\pi^*]\rho)=(w\models[\pi^*]\rho)$$

$\Leftrightarrow$ { By Corollary 1, (56) is valid}

$$(w\models\rho)\cdot(w\models[\pi][\pi^*]\rho)=(w\models[\pi^*]\rho)$$

$\Leftrightarrow$ { defn of $\models$}

$$(w\models\rho\wedge[\pi][\pi^*]\rho)=(w\models[\pi^*]\rho)$$

The proof step annotated with $\star$ comes from

$\prod_{w'\in W}\big((\mathcal{A}_1(w,w')\to(w'\models\rho)\big)$	
$\Leftrightarrow$ { $\mathcal{A}_1$ defn.}	$\Leftrightarrow$ { by (19)}
$(1\to(w\models\rho))\cdot\prod_{w'\in W\setminus\{w\}}\big(\bot\to(w'\models\rho)\big)$	$(w\models\rho)\cdot\top$
$\Leftrightarrow$ { by Lemma 2}	$\Leftrightarrow$ { since $a\le\top$}
$(w\models\rho)\cdot\prod_{w'\in W\setminus\{w\}}(\top)$	$(w\models\rho)$

Therefore, since by (54), $\mathcal{A}_{1+\pi;\pi^*}(w,w')=\mathcal{A}_{\pi^*}(w,w')$ for any $w,w'\in W$, we have $(w\models\rho\wedge[\pi][\pi^*]\rho)=(w\models[\pi^*]\rho)$. By Lemma 3, $\rho\wedge[\pi][\pi^*]\rho\leftrightarrow[\pi^*]\rho$ is valid.

Property **(10.2)** is proved as follows:

$$(w\models[\pi^*](\rho\to[\pi]\rho))$$

$=$ { defn of $\models$}

$$\prod_{w'\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to((w'\models\rho)\to(w'\models[\pi]\rho)\big)$$

$=$ { defn of $\models$}

$$\prod_{w'\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to((w'\models\rho)\to(\prod_{w''\in W}(\mathcal{A}_\pi(w',w'')\to(w''\models\rho)))\big)$$

$=$ { (27) twice }

$$\prod_{w'\in W}\prod_{w''\in W}\big(\mathcal{A}_{\pi^*}(w,w')\to((w'\models\rho)\to(\mathcal{A}_\pi(w',w'')\to(w''\models\rho)))\big)$$

$=$ { (40) 3$\times$ and ;-commutativity from $\mathbb{H}$}

$$\prod_{w'\in W}\prod_{w''\in W}\big((w'\models\rho)\to((\mathcal{A}_{\pi^*}(w,w');\mathcal{A}_\pi(w',w''))\to(w''\models\rho))\big)$$

$\le$ { step $\star\star$ + (35)}

$$
\begin{aligned}
& \prod_{w' \in W} \prod_{w'' \in W} \big((w' \models \rho) \to (\mathcal{A}_{\pi^*}(w, w'') \to (w'' \models \rho))\big) \\
= \quad & \{ \text{by (27)} \} \\
& \prod_{w' \in W} \big((w' \models \rho) \to (\prod_{w'' \in W} (\mathcal{A}_{\pi^*}(w, w'') \to (w'' \models \rho)))\big) \\
= \quad & \{ \text{defn of } \models \} \\
& \prod_{w' \in W} \big((w' \models \rho) \to (w \models [\pi^*]\rho)\big) \\
\leq \quad & \{ \text{infimum} \} \\
& (w \models \rho) \to (w \models [\pi^*]\rho) \\
= \quad & \{ \text{defn of } \models \} \\
& (w \models \rho \to [\pi^*]\rho)
\end{aligned}
$$

Step $\star\star$ holds by definition of program interpretation and (37):

$$
\begin{aligned}
& \mathcal{A}_{\pi^*}(w, w'); \mathcal{A}_{\pi}(w', w'') \leq \sum_{w''' \in W} \big(\mathcal{A}_{\pi^*}(w, w'''); \mathcal{A}_{\pi}(w''', w'')\big) = \mathcal{A}_{\pi^*;\pi}(w, w'') \\
\Rightarrow \quad & \{ \text{by (39)} \} \\
& \mathcal{A}_{\pi^*;\pi}(w, w'') \to (w'' \models \rho) \leq \mathcal{A}_{\pi^*}(w, w'); \mathcal{A}_{\pi}(w', w'') \to (w'' \models \rho) \\
\Rightarrow \quad & \{ \text{by (38)} \} \\
& (w' \models \rho) \to (\mathcal{A}_{\pi^*;\pi}(w, w'') \to (w'' \models \rho)) \\
& \leq (w' \models \rho) \to (\mathcal{A}_{\pi^*}(w, w'); \mathcal{A}_{\pi}(w', w'') \to (w'' \models \rho)) \qquad \square
\end{aligned}
$$

3.2.3. Tests

This section introduces a notion of test in arbitrary $\mathbb{H}$-dynamisations. In order to capture typical computational patterns, for example **while** cycles or **if-then-else** statements, dynamic logics are endowed with an additional kind of program called test, written as $\varphi?$, for φ a formula. This leads us to Kleene algebras with tests (KAT) ([23]), whose syntax is generated by the following grammar:

$$\pi \ni \pi_0 \mid \pi; \pi \mid \pi + \pi \mid \pi^* \mid ?\varphi$$

for $\pi_0 \in \Pi$ and $\varphi \in \mathrm{Fm}^{\mathcal{GDL}(\mathbf{A})}(\Pi, \mathrm{Prop})$. The interpretation of these atomic programs in the standard setting is given by co-reflexive relations $R_{?\varphi} = \{(w, w) | w \models \varphi\}$. In the generic setting of the present work the interpretation of tests is as follows:

$$\mathcal{A}_{?\varphi}(w, w') = \begin{cases} (w \models \varphi) & \text{if } w = w' \\ \bot & \text{otherwise} \end{cases}$$

Lemma 11. *Let* **A** *be a complete* $\mathbb{H}$*-action lattice. The following are valid formulæ in* $\mathcal{GDL}(\mathbf{A})$*:*

(11.1) $\langle ?\psi \rangle \varphi \leftrightarrow (\psi \wedge \varphi)$
(11.2) $[?\psi]\varphi \leftrightarrow (\psi \to \varphi)$

Proof. Proof of **(11.1)**:

$$
\begin{aligned}
& (w \models \langle ?\psi \rangle \varphi) \\
= \quad & \{ \text{defn of } \models \} \\
& \textstyle\sum_{w' \in W} \big(\mathcal{A}_{?\psi}(w, w'); (w' \models \varphi)\big) \\
= \quad & \{ w \neq w' \Rightarrow \mathcal{A}_{?\psi}(w, w') = \bot + (4) + (9) \} \\
& (w \models \psi); (w \models \varphi)
\end{aligned}
\qquad
\begin{aligned}
= \quad & \{ \text{by (47)} \} \\
& (w \models \psi) \cdot (w \models \varphi) \\
= \quad & \{ \text{defn of } \models \} \\
& (w \models \psi \wedge \varphi)
\end{aligned}
$$

Proof of **(11.2)**:

$$
\begin{array}{ll}
 & (w \models [?\psi]\varphi) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \prod_{w' \in W} \left(\mathcal{A}_{?\psi}(w, w') \to (w' \models \varphi) \right) \\
= & \quad \{ \text{ defn. of } \mathcal{A}_{?\psi} + (17) \} \\
 & \prod_{w' \in W - \{w\}} \left(0 \to (w' \models \varphi) \right) \cdot \\
 & ((w \models \psi) \to (w \models \varphi)) \\
= & \quad \{ \text{ by Lemma 2} \}
\end{array}
\quad \Bigg| \quad
\begin{array}{ll}
 & \prod_{w' \in W \setminus \{w\}} \{\top\} \cdot \\
 & ((w \models \psi) \to (w \models \varphi)) \\
= & \quad \{ a \leq \top \Leftrightarrow a \cdot \top = a \} \\
 & (w \models \psi) \to (w \models \varphi) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \psi \to \varphi)
\end{array}
\quad \square
$$

3.2.4. Negations

As action lattices have a least element $\bot$, the usual negation

$$\neg\rho =^{def} (\rho \to \bot) \tag{59}$$

can be defined. However, the usual correspondence expressed by $[\pi]\rho \leftrightarrow \neg\langle\pi\rangle\neg\rho$ it is not generally valid, just holding in a restricted subclass of dynamisations. The same happens with some classical properties, for example involution $\neg\neg x \leftrightarrow x$.

Lemma 12. *Let* **A** *be a complete* $\mathbb{I}$*-action lattice. Then*

$$\neg\langle\pi\rangle\rho \leftrightarrow [\pi]\neg\rho \tag{60}$$

is valid in $\mathcal{GDL}(\mathbf{A})$.

Proof.

$$
\begin{array}{ll}
 & (w \models \neg\langle\pi\rangle\rho) \\
= & \quad \{ \text{ by (59)} \} \\
 & (w \models (\langle\pi\rangle\rho \to \bot) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models \langle\pi\rangle\rho) \to (w \models \bot) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \left(\sum_{w' \in W} \left(\mathcal{A}_\pi(w, w'); \right. \right. \\
 & \left. \left. (w' \models \rho) \right) \right) \to (w \models \bot)
\end{array}
\quad \Bigg| \quad
\begin{array}{ll}
= & \quad \{ \text{ by (28) and } (w' \models \bot) = \bot = (w \models \bot) \} \\
 & \prod_{w' \in W} \left(\mathcal{A}_\pi(w, w') \to \right. \\
 & \left. ((w' \models \rho) \to (w' \models \bot)) \right) \\
= & \quad \{ \text{ defn of } \models \} \\
 & \prod_{w' \in W} \left(\mathcal{A}_\pi(w, w') \to (w' \models \rho \to \bot) \right) \\
= & \quad \{ \text{ defn of } \models \} \\
 & (w \models [\pi](\rho \to \bot)) \\
= & \quad \{ \text{ by (59)} \} \\
 & (w \models [\pi]\neg\rho)
\end{array}
$$

Therefore, we conclude the validity of (60) by Lemma 3. $\square$

The following result establishes a sufficient condition for the usual duality between box and diamond modalities. For that, let us consider the following direct consequence of (48):

Lemma 13. *The property*

$$(a \to \bot) \to \bot = a \tag{61}$$

holds in any $\mathbb{MV}$*-action lattice.*

Corollary 2. *Let* **A** *be a* $\mathbb{MV}$*-action lattice. Then,*

$$[\pi]\rho \leftrightarrow \neg\langle\pi\rangle\neg\rho \tag{62}$$

is valid in $\mathcal{GDL}(\mathbf{A})$.

Proof. By Lemma 13, $\neg\neg x = x$. By Lemma 3, $\neg\neg\rho \leftrightarrow \rho$ is valid in any $\mathbb{MV}$-dynamisation. Moreover, by (60), we have that $\neg\langle\pi\rangle\rho \leftrightarrow [\pi]\neg\rho$ is valid. Thus, $\neg\langle\pi\rangle(\neg\rho) \leftrightarrow [\pi]\neg(\neg\rho) \leftrightarrow [\pi]\rho$ is valid. $\square$

In order to illustrate a case of a non-$\mathbb{MV}$-algebra where the property (62) fails, let us consider the action $\mathbb{H}$-lattice **3** of Example 2, a model $\mathcal{A}$ with $W = \{\bullet\}$, a program π such that $\mathcal{A}_\pi(\bullet,\bullet) = \top$ and with a proposition p such that $V(p,\bullet) = u$. Then, on the one hand we have:

$$\begin{aligned}(\bullet \models [\pi]p) &= \mathcal{A}_\pi(\bullet,\bullet) \to (\bullet \models p)\\ &= \top \to u\\ &= u\end{aligned}$$

On the other hand:

$$\begin{aligned}(\bullet \models \neg\langle\pi\rangle\neg p) &= (\bullet \models (\langle\pi\rangle\neg p) \to \bot)\\ &= (\bullet \models \langle\pi\rangle(p \to \bot)) \to \bot\\ &= (\mathcal{A}_\pi(\bullet,\bullet) \wedge (\bullet \models p \to \bot)) \to \bot\\ &= (\mathcal{A}_\pi(\bullet,\bullet) \wedge (u \to \bot)) \to \bot\\ &= (\top \wedge \bot) \to \bot\\ &= \bot \to \bot\\ &= \top\end{aligned}$$

Hence, $(\bullet \models [\pi]p) \neq (\bullet \models \neg\langle\pi\rangle\neg p)$.

The study of the diamond version of Segerberg axiom, *i.e.*, the axiom $\langle\pi^*\rangle\rho \to \langle\pi^*\rangle(\neg\rho \wedge \langle\pi\rangle\rho)$, entails the need for some auxiliary developments:

Lemma 14. *The following properties hold in any structure that is both a $\mathbb{H}$ and $\mathbb{MV}$-action lattice:*

$$(a \to c) + (b \to c) = (a;b) \to c \tag{63}$$

$$(a \to b) = (a \to \bot) + b \tag{64}$$

$$(a \to b) = (b \to \bot) \to (a \to \bot) \tag{65}$$

$$(a + b) = ((a \to \bot) \cdot (b \to \bot)) \to \bot \tag{66}$$

Proof. In order to prove (63), observe that, since $a \leq a$ and $b \leq 1$, we have, by (42), $a;b \leq a$ and, analogously, $a;b \leq b$. By (39), we have also $a \to c \leq (a;b) \to c$ and $b \to c \leq (a;b) \to c$. By (33) and (3),

$$(a \to c) + (b \to c) \leq (a;b) \to c \tag{67}$$

For the other inequality, observe that

$$\begin{aligned}&(a \to c) + (b \to c)\\ =\ &\{\text{(48) and (2)}\}\\ &\big((b \to c) \to (a \to c)\big) \to (a \to c)\\ =\ &\{\text{(40)}\}\\ &\big((a;(b \to c)) \to c\big) \to (a \to c)\\ =\ &\{\text{(40)}\}\\ &\Big(a;\big((a;(b \to c)) \to c\big)\Big) \to c\\ =\ &\{\text{(40)}\}\\ &a \to \Big(\big((a;(b \to c)) \to c\big) \to c\Big)\\ =\ &\{\text{(40)}\}\\ &a \to \Big(a \to \big(((b \to c) \to c) \to c\big)\Big)\end{aligned}$$

$$\begin{aligned}=\ &\{\text{(48)}\}\\ &a \to \Big(a \to \big((b \to c) + c\big)\Big)\\ =\ &\{\text{(40)}\}\\ &(a;a) \to \big((b \to c) + c\big)\\ =\ &\{\text{(47) and (19)}\}\\ &a \to \big((b \to c) + c\big)\\ \geq\ &\{\text{(38)}\}\\ &a \to (b \to c)\\ =\ &\{\text{(40)}\}\\ &(a;b) \to c\end{aligned}$$

Proof of (64),

$$\begin{array}{ll|ll} & (a \to \bot) + b & & \\ = & \quad \{\,(61)\,\} & = & \quad \{\,(40)\,\} \\ & (a \to \bot) + ((b \to \bot) \to \bot) & & a \to ((b \to \bot) \to \bot) \\ = & \quad \{\,(63)\,\} & = & \quad \{\,(61)\,\} \\ & (a;(b \to \bot)) \to \bot & & a \to b \end{array}$$

Proof of (65)

$$\begin{array}{ll|ll} & (b \to \bot) \to (a \to \bot) & = & \quad \{\,(40)\,\} \\ = & \quad \{\,(40)\,\} & & a \to ((b \to \bot) \to \bot) \\ & (a;(b \to \bot)) \to \bot & = & \quad \{\,(61)\,\} \\ & & & a \to b \end{array}$$

In order to prove (66), observe that

$$\begin{array}{ll|ll} & \big((a \to \bot) \cdot (b \to \bot)\big) \to \bot & & \\ = & \quad \{\,(28)\,\} & = & \quad \{\,(61)\,\} \quad \square \\ & \big((a + b) \to \bot\big) \to \bot & & a + b \end{array}$$

Lemma 15. *Let* **A** *be* $\mathbb{MV}$*-action lattice that is also a* $\mathbb{H}$*-action lattice. Then*

(14.1) $\langle \pi^* \rangle \rho \to \langle \pi^* \rangle (\neg \rho \wedge \langle \pi \rangle \rho)$

is valid in $\mathcal{GDL}(\mathbf{A})$.

Proof. From Lemma 10 we have that, for any $w \in W$, $(w \models [\pi^*](\rho \to [\pi]\rho) \to (\rho \to [\pi^*]\rho))$. Moreover, applying the definition of $\models$ twice, leads to

$$\begin{array}{ll} & (w \models [\pi^*](\rho \to [\pi]\rho)) \to ((w \models \rho) \to (w \models [\pi^*]\rho)) \\ = & \quad \{\,(40) \text{ and } (47)\,\} \\ & \big((w \models \rho) \cdot (w \models [\pi^*](\rho \to [\pi]\rho))\big) \to (w \models [\pi^*]\rho) \end{array}$$

In particular, for the case $\neg\rho$,

$$\begin{array}{ll} & \big((w \models \neg\rho) \cdot (w \models [\pi^*](\neg\rho \to [\pi]\neg\rho))\big) \to (w \models [\pi^*]\neg\rho) \\ = & \quad \{\,(60)\,\} \\ & \big((w \models \neg\rho) \cdot (w \models [\pi^*](\neg\rho \to \neg\langle\pi\rangle\rho))\big) \to (w \models \neg\langle\pi^*\rangle\rho) \\ = & \quad \{\,\models \text{defn} + (\star :=)\ (w \models \neg\rho) = (w \models \rho) \to (w \models \bot) = \neg(w \models \rho)\,\} \\ & \Big(\neg(w \models \rho) \cdot \prod_{w' \in W} \big(\mathcal{A}_{\pi^*}(w, w') \to (\neg(w' \models \rho) \to \neg(w' \models \langle\pi\rangle\rho))\big)\Big) \to \neg(w \models \langle\pi^*\rangle\rho) \\ = & \quad \{\,(65) + (64)\,\} \\ & \Big(\neg(w \models \rho) \cdot \prod_{w' \in W} \big(\mathcal{A}_{\pi^*}(w, w') \to \neg(\neg(w' \models \rho) \cdot (w' \models \langle\pi\rangle\rho))\big)\Big) \to \neg(w \models \langle\pi^*\rangle\rho) \\ = & \quad \{\,\models \text{defn}\,\} \\ & \Big(\neg(w \models \rho) \cdot (w \models [\pi^*]\neg(\neg\rho \wedge \langle\pi\rangle\rho))\Big) \to \neg(w \models \langle\pi^*\rangle\rho) \\ = & \quad \{\,(60) + (\star)\,\} \end{array}$$

$$
\begin{array}{ll}
 & \big(\neg(w \models \rho) \cdot \neg(w \models \langle\pi^*\rangle(\neg\rho \wedge \langle\pi\rangle\rho))\big) \to \neg(w \models \langle\pi^*\rangle\rho) \\
= & \quad \{\ (66)\} \\
 & \neg\big((w \models \rho) + (w \models \langle\pi^*\rangle(\neg\rho \wedge \langle\pi\rangle\rho))\big) \to \neg(w \models \langle\pi^*\rangle\rho) \\
= & \quad \{\ (65)\} \\
 & (w \models \langle\pi^*\rangle\rho) \to ((w \models \rho) + (w \models \langle\pi^*\rangle(\neg\rho \wedge \langle\pi\rangle\rho)))\big) \\
= & \quad \{\ \text{defn of } \models\} \\
 & (w \models \langle\pi^*\rangle\rho \to (\rho \vee \langle\pi^*\rangle(\neg\rho \wedge \langle\pi\rangle\rho)) \qquad \square
\end{array}
$$

3.3. Inference rules

In this subsection we discuss the soundness of the inference rules *modus ponens*,

$$\frac{\rho \quad \rho \to \rho'}{\rho'} \tag{68}$$

and *necessity*

$$\frac{\rho}{[\pi]\rho} \tag{69}$$

Lemma 16. *Both* modus ponens *and* necessity *are sound for* $\mathbb{I}$*-dynamisations.*

Proof. In order to prove the soundness of (68), suppose that $\varphi \to \varphi'$ is valid, *i.e.*, that for any $v \in W$, $(v \models \varphi \to \psi) = \top$, *i.e.*, $(v \models \varphi) \to (v \models \psi) = \top$. By 5. of Lemma 2, this is equivalent to $(v \models \varphi) \leq (v \models \psi)$. Suppose also that φ is valid, *i.e.*, that for any $v \in W$, $(v \models \varphi) = \top$. Hence $\top \leq (v \models \psi)$, *i.e.*, $(v \models \psi) = \top$.

In order to prove the soundness of (69), let us suppose φ valid, *i.e.*, that for any $v \in W$, $(v \models \varphi) = \top$. Hence, let $w \in W$,

$$
\begin{array}{ll}
 & (w \models [\pi]\varphi) \\
= & \quad \{\ \text{programs' interpretation}\} \\
 & \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to (w' \models \varphi)\big) \\
= & \quad \{\ \text{by hypothesis } (w \models \varphi) = \top\}
\end{array}
\qquad
\begin{array}{ll}
 & \prod_{w' \in W} \big(\mathcal{A}_\pi(w, w') \to \top\big) \\
= & \quad \{\ \text{by (30)}\} \\
 & \prod_{w' \in W} \top \\
= & \quad \{\ \text{by (19)}\} \\
 & \top
\end{array}
\qquad \square
$$

4. Conclusion and future work

This paper is essentially an exercise that aims at the identification of the minimal set of properties required to generate a well behaved many-valued (propositional) dynamic logic. This extends our previous work [29] in a number of directions, namely, the explicit consideration of [_]-modalities, negations and tests. This is, however, the beginning of a more ambitious project on the systematic generation of dynamic logics, their theory and tool support. The following notes describe some of the next steps.

A main objective of this exercise was to understand *how dynamic dynamisations are*. Our approach took the standard axiomatics of PDL as a starting point for a systematic study of some of its fragments with respect to particular dynamisation classes. This lead to the identification of $\mathbb{I}$-dynamisations as a class of many-valued dynamic logics sound for the positive existential fragment of PDL without tests, *i.e.*, for those axioms not involving box modalities, negations or tests. Its subclass of $\mathbb{H}$-dynamisations was identified as the class of many-valued dynamic logics sound for the positive axiomatisation of PDL. Finally, the class of $\mathbb{MV}$-dynamisations, satisfying an involutive negation, was identified as a class of dynamisations sound for the whole axiomatisation of PDL.

An axiomatics for PDL was originally introduced by Segerberg in [41], with its completeness further discussed by Kozen and Parikh in [24] (see also [16,17]). The calculus consists of the axioms recalled in Lemmas 7–10 together with the second axiom of Lemma 11 and the inference rules presented in the Subsection 3.3. The reader may be wondering about the potential redundancy of this exercise, by proving properties that are presumably derivable from these calculi. Actually, this is not the case: as discussed above, in the general case, we do not have involutive negations and, hence, the standard duality between $\langle\_\rangle$ and $[\_]$ modalities fails. Moreover, as already discussed, the soundness of [_]-axioms is only proven for $\mathbb{H}$-dynamisations. This context justifies the study of axioms for both modalities, instead of assuming just one of them.

Therefore, the characterisation of specific classes of dynamisations whose semantics is complete for specific fragments of this axiomatics emerges as an interesting topic for future work.

The exercise remains, however, incomplete. Actually, there is a number of well known action lattices that do not belong to any of these classes. Such are the cases of the non-integral action lattices of relational algebra over a set, and of languages over a finite alphabet, introduced in Examples 6 and 7, as well as, the action lattice based in Warshall algebra discussed in Example 5. A proper answer to the question above has to be given on a case-by-case basis. In this process, the study of these less conventional examples can be of potential interest. For instance, a variant of the dynamisation based in the action lattice of Example 5 was already approached by the authors as an interesting formalism to deal with resource dependent systems [28]. In this case, the evaluation of a program or, more generally, of a given behaviour, corresponds to the costs of its execution (expressed by a natural number). Moreover, the truth degree of a formula, corresponds to the amount of resources needed for validate the corresponding property.

It is well known that both in PDL, and in its algebraic counterpart KAT, the standard imperative programming constructors can be formalised using tests. First of all, we note that the essence of a test in generic $\mathcal{GDL}(\mathbf{A})$ logics, with $\mathbf{A}$ an $\mathbb{H}$-action lattice, is quite distinct from its meaning in PDL (and KAT). While the latter is a kind of switch among execution paths, the former can be understood as an operator to weigh choices in transitions.

In particular, although able to capture the essence of the **if_then** combinator through $(?\varphi;\pi)$ the interpretation of **if_then_else** in PDL by the program $(?\varphi;\pi) + (?(\neg\varphi);\pi')$ clearly differs from the usual interpretation. Instead of a "disjuntive" choice of paths, we have a weighted one. An interesting line of research emerges: how can we accommodate the standard imperative programming commands in this 'weighted' setting?

Having introduced a generic method to build propositional many-valued dynamic logics, parameterised by an action lattice (intended to capture the nature of computations), we plan to consider a new level of parameterisation – the choice of the logic in which properties can be expressed. I.e., instead of considering propositions as atomic sentences to talk about computational properties, we would like to build logics with increased expressiveness, *e.g.* equational, first, or even higher order, etc. Such a plan may develop in a way similar to what happened in hybridisation of logics [30,9,27]. This is a process to build the characteristic features of hybrid logic [3], and the corresponding modal semantics, on top of whatever logic is found appropriate as a basic specification formalism. As instantiations of this method, a wide range of existing hybrid logics can be captured and new ones were found.

Acknowledgements

The joint work with Luís S. Barbosa in several related topics, including the preliminary work about dynamisation (reported in [29]), as well as productive discussions in the preparation of this paper, were determinant for achieving the reported results. We are very grateful to him. We also acknowledge Isabel Ferreirim and Félix Bou for the kind explanation of specific points of their work. Finally, we would like to mention that, the careful work of the anonymous reviewers improved, in many points, the quality of the present paper.

This work is funded by ERDF – European Regional Development Fund, through the COMPETE Programme, and by National Funds through FCT within project DaLí – Dynamic logics for cyber-physical systems: towards contract based design PTDC/EEI-CTP/4836/2014. A. Madeira and R. Neves are supported by the FCT grants SFRH/BPD/103004/2014 and SFRH/BD/52234/2013, respectively. Finally, M. Martins is supported by FCT project UID/MAT/04106/2013 at CIDMA and the EU FP7 Marie Curie PIRSES-GA-2012-318986 project GeTFun: Generalizing Truth-Functionality.

References

[1] L.S. Barbosa, J.N. Oliveira, A.M. Silva, Calculating invariants as coreflexive bisimulations, in: J. Meseguer, G. Rosu (Eds.), Algebraic Methodology and Software Technology, Proceedings of the 12th International Conference, AMAST 2008, Urbana, IL, USA, July 28–31, 2008, in: Lecture Notes in Computer Science, vol. 5140, Springer, 2008, pp. 83–99.
[2] B. Beckert, A dynamic logic for the formal verification of Java card programs, in: I. Attali, T.P. Jensen (Eds.), Java Card Workshop, in: Lecture Notes in Computer Science, vol. 2041, Springer, 2000, pp. 6–24.
[3] P. Blackburn, Representation, reasoning, and relational structures: a hybrid logic manifesto, Log. J. IGPL 8 (3) (2000) 339–365.
[4] W.J. Blok, I.M.A. Ferreirim, On the structure of hoops, Algebra Univers. 43 (2–3) (2000) 233–257.
[5] F. Bou, F. Esteva, L. Godo, R.O. Rodríguez, Characterizing fuzzy modal semantics by fuzzy multimodal systems with crisp accessibility relations, in: J.P. Carvalho, D. Dubois, U. Kaymak, J.M. da Costa Sousa (Eds.), Proceedings of the Joint 2009 International Fuzzy Systems Association World Congress and 2009 European Society of Fuzzy Logic and Technology Conference, Lisbon, Portugal, 20–24 July, 2009, pp. 1541–1546.
[6] F. Bou, F. Esteva, L. Godo, R.O. Rodríguez, On the minimum many-valued modal logic over a finite residuated lattice, J. Log. Comput. 21 (5) (2011) 739–790.
[7] R. Cignoli, I. D'Ottaviano, D. Mundici, Algebraic Foundations of Many-Valued Reasoning, Kluwer Academic Publishers, 1999.
[8] J.H. Conway, Regular Algebra and Finite Machines, 1971, Printed in GB by William Clowes & Sons Ltd.
[9] R. Diaconescu, A. Madeira, Encoding hybridized institutions into first-order logic, Math. Struct. Comput. Sci. FirstView (2014) 1–44, http://dx.doi.org/10.1017/S0960129514000383.
[10] M. Droste, P. Gastin, Weighted automata and weighted logics, Theor. Comput. Sci. 380 (1–2) (2007) 69–86.
[11] M. Fitting, Many-valued modal logics, Fundam. Inform. 15 (3–4) (1991) 235–254.
[12] M. Fitting, Many-valued model logics II, Fundam. Inform. 17 (1–2) (1992) 55–73.
[13] H. Furusawa, The categories of Kleene algebras, action algebras and action lattices are related by adjunctions, in: R. Berghammer, B. Möller, G. Struth (Eds.), RelMiCS, in: Lecture Notes in Computer Science, vol. 3051, Springer, 2003, pp. 124–136.
[14] N. Galatos, P. Jipsen, T. Kowalski, H. Ono, Residuated Lattices: An Algebraic Glimpse at Substructural Logics, Elsevier, 2007.

[15] S. Gottwald, A Treatise on Many-Valued Logics, Studies in Logic and Computation, vol. 9, Research Studies Press, 2001.
[16] D. Harel, D. Kozen, J. Tiuryn, Dynamic logic, in: Handbook of Philosophical Logic, MIT Press, 1984, pp. 497–604.
[17] D. Harel, D. Kozen, J. Tiuryn, Dynamic Logic, MIT Press, 2000.
[18] J. Hughes, A.C. Esterline, B. Kimiaghalam, Means-end relations and a measure of efficacy, J. Log. Lang. Inf. 15 (1–2) (2006) 83–108.
[19] P. Jipsen, C. Tsinakis, A survey of residuated lattices, in: J. Martńez (Ed.), Ordered Algebraic Structures, in: Developments in Mathematics, vol. 7, Springer US, 2002, pp. 19–56.
[20] D. Kozen, A probabilistic PDL, J. Comput. Syst. Sci. 30 (2) (1985) 162–178.
[21] D. Kozen, On action algebras, in: Logic and Flow of Information, Amsterdam, 1991.
[22] D. Kozen, A completeness theorem for Kleene algebras and the algebra of regular events, Inf. Comput. 110 (2) (1994) 366–390.
[23] D. Kozen, Kleene algebra with tests, ACM Trans. Program. Lang. Syst. 19 (3) (1997) 427–443.
[24] D. Kozen, R. Parikh, An Elementary Proof of the Completness of PDL, Theor. Comput. Sci. 14 (1981) 113–118.
[25] C. Liau, Many-valued dynamic logic for qualitative decision theory, in: N. Zhong, A. Skowron, S. Ohsuga (Eds.), New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, Proceedings of the 7th International Workshop, RSFDGrC '99, Yamaguchi, Japan, November 9–11, 1999, in: Lecture Notes in Computer Science, vol. 1711, Springer, 1999, pp. 294–303.
[26] B. Lopes, M.R.F. Benevides, E.H. Haeusler, Propositional dynamic logic for Petri nets, Log. J. IGPL 22 (5) (2014) 721–736.
[27] A. Madeira, Foundations and techniques for software reconfigurability, Ph.D. thesis, Universidades do Minho, Aveiro and Porto, 2013 (Joint MAP-i Doctoral Programme).
[28] A. Madeira, R. Neves, M.A. Martins, L.S. Barbosa, A logic for robotics?, AIP Conf. Proc. 1648 (1) (2015).
[29] A. Madeira, R. Neves, M.A. Martins, L.S. Barbosa, A dynamic logic for every season, in: C. Braga, N. Martí-Oliet (Eds.), Formal Methods: Foundations and Applications, Proceedings of the 17th Brazilian Symposium, SBMF 2014, Maceió, AL, Brazil, September 29–October 1, 2014, in: Lecture Notes in Computer Science, vol. 8941, Springer, 2015, pp. 130–145.
[30] M.A. Martins, A. Madeira, R. Diaconescu, L.S. Barbosa, Hybridization of institutions, in: A. Corradini, B. Klin, C. Cîrstea (Eds.), Algebra and Coalgebra in Computer Science, CALCO 2011, Winchester, UK, August 30–September 2, 2011, in: Lecture Notes in Computer Science, vol. 6859, Springer, 2011, pp. 283–297.
[31] A. Mironov, Fuzzy modal logics, J. Math. Sci. 128 (6) (2005) 3461–3483.
[32] O. Mürk, D. Larsson, R. Hähnle, Key-c: a tool for verification of c programs, in: F. Pfenning (Ed.), CADE, in: Lecture Notes in Computer Science, vol. 4603, Springer, 2007, pp. 385–390.
[33] N.J. Nilsson, Probabilistic logic, Artif. Intell. 28 (1) (1986) 71–87.
[34] J.N. Oliveira, Extended static checking by calculation using the pointfree transform, in: A. Bove, L.S. Barbosa, A. Pardo, J.S. Pinto (Eds.), Language Engineering and Rigorous Software Development, International LerNet ALFA Summer School 2008, Piriapolis, Uruguay, February 24–March 1, 2008, Revised Tutorial Lectures, in: Lecture Notes in Computer Science, vol. 5520, Springer, 2009, pp. 195–251.
[35] J.N. Oliveira, Weighted automata as coalgebras in categories of matrices, Int. J. Found. Comput. Sci. 24 (6) (2013) 709–728.
[36] J.N. Oliveira, A relation-algebraic approach to the "hoare logic" of functional dependencies, J. Log. Algebr. Meth. Program. 83 (2) (2014) 249–262.
[37] M. Pauly, R. Parikh, Game logic – an overview, Stud. Log. 75 (2) (2003) 165–182.
[38] A. Platzer, Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics, Springer, 2010.
[39] A. Platzer, A complete axiomatization of quantified differential dynamic logic for distributed hybrid systems, Log. Methods Comput. Sci. 8 (4) (2012).
[40] V.R. Pratt, Action logic and pure induction, in: JELIA, in: Lecture Notes in Computer Science, vol. 478, Springer, 1990, pp. 97–120.
[41] K. Segerberg, A completeness theorem in the modal logic of programs, Not. Am. Math. Soc. 24 (6) (1977) 31–46.